#3

Docket No. 1075.1137/JDH

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:

Masayoshi HASHIMA, et al.

Serial No.: New

Filed: January 3, 2001

Group Art Unit: Unassigned

Examiner: Unassigned

For: **SUPPORT SYSTEM, AND COMPUTER-READABLE RECORDING MEDIUM IN WHICH SUPPORT PROGRAM IS RECORDED**

SUBMISSION OF CERTIFIED COPY OF PRIOR FOREIGN APPLICATION IN ACCORDANCE WITH THE REQUIREMENTS OF 37 C.F.R. §1.55

Honorable Commissioner of
Patents and Trademarks
Washington, D.C. 2023l

Sir:

In accordance with the provisions of 37 C.F.R. §1.55, the applicant(s) submit(s) herewith a certified copy of the following foreign application:

Japanese Patent Application No. 2000-034642
Filed: February 14, 2000.

It is respectfully requested that the applicants be given the benefit of the foreign filing date as evidenced by the certified papers attached hereto, in accordance with the requirements of 35 U.S.C. §119.

Respectfully submitted,
STAAS & HALSEY LLP

Date: January 3, 2001

By: _____
James D. Halsey, Jr.
Registration No. 22,729

700 Eleventh Street, N.W., Suite 500
Washington, D.C. 20001
(202) 434-1500

日　本　国　特　許　庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２０００年　２月１４日

出願番号
Application Number: 特願２０００－０３４６４２

出願人
Applicant (s): 富士通株式会社

２０００年　９月２９日

特許庁長官
Commissioner,
Patent Office 及川耕造

出証番号　出証特２０００－３０８０２４８

【書類名】　特許願

【整理番号】　9951877

【提出日】　平成12年 2月14日

【あて先】　特許庁長官殿

【国際特許分類】　G06F 17/50

【発明の名称】　支援システムおよび組込み支援プログラムを記録したコンピュータ読み取り可能な記録媒体

【請求項の数】　18

【発明者】

【住所又は居所】　神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

【氏名】　橋間　正芳

【発明者】

【住所又は居所】　神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

【氏名】　千田　陽介

【発明者】

【住所又は居所】　神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

【氏名】　佐藤　裕一

【特許出願人】

【識別番号】　000005223

【氏名又は名称】　富士通株式会社

【代理人】

【識別番号】　100092978

【弁理士】

【氏名又は名称】　真田　有

【電話番号】　0422-21-4222

【手数料の表示】

【予納台帳番号】　007696

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9704824

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　支援システムおよび組込み支援プログラムを記録したコンピュータ読み取り可能な記録媒体

【特許請求の範囲】

【請求項１】　アクチュエータおよびセンサを含む複数の部品からなる機構を３次元的に設計する機構設計部と、

該機構を３次元機構モデルとして内部に構築され該機構の動作をシミュレートする３次元機構モデルシミュレーション部と、

該機構に組み込まれ該機構の動作を制御するための制御プログラムを組込みソフトウェアとして開発する組込みソフトウェア開発部と、

該機構設計部による設計データを、該３次元機構モデルに反映させるべく該機構設計部から該３次元機構モデルシミュレーション部へ入力する第１インタフェース部と、

該３次元機構モデルシミュレーション部の動作と該組込みソフトウェア開発部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該組込みソフトウェア開発部との間でデータ送受を行なう第２インタフェース部とをそなえて構成されていることを特徴とする、支援システム。

【請求項２】　該第１インタフェース部が、該３次元機構モデルシミュレーション部によるシミュレーション結果を、該機構の設計に反映させるべく該３次元機構モデルシミュレーション部から該機構設計部へ入力することを特徴とする、請求項１記載の支援システム。

【請求項３】　該組込みソフトウェア開発部が、該組込みソフトウェアの詳細設計を行なうべく該組込みソフトウェアの仕様を記述する状態遷移図・表を作成・編集する状態遷移図・表作成部を含んで構成され、

該第２インタフェース部が、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該状態遷移図・表作成部との間でデータ送受を行なうことを特徴とする、請求項１または請求項２に記載の支援システム。

【請求項４】　該状態遷移図・表作成部が、複数のタスクを並行して実行す

るマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行し、

該第２インタフェース部が、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させることを特徴とする、請求項３記載の支援システム。

【請求項５】　該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させることを特徴とする、請求項４記載の支援システム。

【請求項６】　該組込みソフトウェア開発部が、開発中の該組込みソフトウェアを組み込まれたマイクロコンピュータチップを含んで構成され、

該第２インタフェース部が、該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作との同期をとりながら該３次元機構モデルシミュレーション部と該マイクロコンピュータチップとの間でデータ送受を行なうことを特徴とする、請求項１～請求項５のいずれか１項に記載の支援システム。

【請求項７】　該マイクロコンピュータチップが、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行し、

該第２インタフェース部が、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させることを特徴とする、請求項６記載の支援システム。

【請求項８】　該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させることを特徴とする、請求項７記載の支援システム。

【請求項９】　該第２インタフェース部が、

該組込みソフトウェア開発部から該３次元機構モデルシミュレーション部へ、該３次元機構モデルにおける該アクチュエータに対するアクチュエータ指令信号を受け渡すとともに、

該３次元機構モデルシミュレーション部から該組込みソフトウェア開発部へ、該アクチュエータ指令信号に応じたシミュレーション結果として得られる、該３次元機構モデルにおける該センサからのセンサ信号を受け渡すことを特徴とする、請求項１～請求項８のいずれか１項に記載の支援システム。

【請求項１０】　該アクチュエータに対する該アクチュエータ指令信号と該３次元機構モデルシミュレーション部からの該センサ信号との時間変化をリアルタイムで解析して表示しうる解析部をそなえて構成されていることを特徴とする、請求項９記載の支援システム。

【請求項１１】　アクチュエータおよびセンサを含む複数の部品からなる機構を制御すべくその機構に制御プログラムとして組み込まれる組込みソフトウェアの開発を支援する機能をコンピュータにより実現するための、支援プログラムを記録したコンピュータ読み取り可能な記録媒体であって、

該支援プログラムとして、

該機構を３次元的に設計する機構設計部として、該コンピュータを機能させる機構設計プログラムと、

該機構を３次元機構モデルとして内部に構築され該機構の動作をシミュレートする３次元機構モデルシミュレーション部として、該コンピュータを機能させる３次元機構モデルシミュレーションプログラムと、

該組込みソフトウェアを開発する組込みソフトウェア開発部として、該コンピュータを機能させる組込みソフトウェア開発プログラムと、

該機構設計部による設計データを、該３次元機構モデルに反映させるべく該機構設計部から該３次元機構モデルシミュレーション部へ入力する第１インタフェース部として、該コンピュータを機能させる第１インタフェースプログラムと、

該３次元機構モデルシミュレーション部の動作と該組込みソフトウェア開発部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該組込みソフトウェア開発部との間でデータ送受を行なう第２インタフェース部として、

該コンピュータを機能させる第２インタフェースプログラムとが記録されていることを特徴とする、支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項１２】　該第１インタフェースプログラムが、該３次元機構モデルシミュレーション部によるシミュレーション結果を、該機構の設計に反映させるべく該３次元機構モデルシミュレーション部から該機構設計部へ入力させるように構成されていることを特徴とする、請求項１記載の支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項１３】　該組込みソフトウェア開発プログラムが、該組込みソフトウェアの詳細設計を行なうべく該組込みソフトウェアの仕様を記述する状態遷移図・表を作成・編集する状態遷移図・表作成部として、該コンピュータを機能させる状態遷移図・表作成プログラムを含むものであり、

該第２インタフェースプログラムが、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該状態遷移図・表作成部との間でデータ送受を行なわせるように構成されていることを特徴とする、請求項１１または請求項１２に記載の支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項１４】　該状態遷移図・表作成プログラムが、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行させるように構成され、

該第２インタフェースプログラムが、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させることを特徴とする、請求項１３記載の支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項１５】　該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させることを特徴とする、請求項１４記載の支援プログラムを記録したコンピュータ読み取り可能な記録

媒体。

【請求項１６】　該第２インタフェースプログラムが、該３次元機構モデルシミュレーション部の動作と、開発中の該組込みソフトウェアを組み込まれたマイクロコンピュータチップの動作との同期をとりながら該３次元機構モデルシミュレーション部と該マイクロコンピュータチップとの間でデータ送受を行なわせるように構成されていることを特徴とする、請求項１１～請求項１５のいずれか１項に記載の支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項１７】　該マイクロコンピュータチップが、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行し、

該第２インタフェースプログラムが、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させることを特徴とする、請求項１６記載の支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【請求項１８】　該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させることを特徴とする、請求項１７記載の支援プログラムを記録したコンピュータ読み取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、アクチュエータやセンサを有し３次元的な動作を行なう機構の設計・開発にかかる技術であって、特に、機構を制御すべくその機構に組み込まれる制御プログラム（組込みソフトウェア）の開発を支援するための、支援システムおよび支援プログラムを記録したコンピュータ読み取り可能な記録媒体に関する。

【０００２】

【従来の技術】

一般に、アクチュエータ（モータ）やセンサを有し３次元的な動作を行なう機構（メカ）を設計する際には、その機構の構想を練った後、詳細設計，出図，部品手配を行なってから、部品の組立を行なって実機を試作し、実機の動作等についての評価を行なう。そして、評価の結果に応じて設計変更を行なってから設計変更後の実機を試作し、再び評価を行なうという処理を繰り返し、評価の結果が良好であれば、設計を完了する。なお、ここでいう機構としては、例えばＣＤチェンジャ，ＭＤチェンジャ，プリンタ，マニピュレータ等が挙げられる。

【０００３】

また、一般に、上述のごとく設計された機構を動作させるべく、その機構を制御するための制御プログラムを開発し、その制御プログラムを、制御対象の機構内に組み込まれるマイクロコンピュータ（以下、マイコンという場合がある）で実行させるようにしている。このマイコンが実行する上記制御プログラムのことを、以下、組込みソフトウェアという場合がある。

【０００４】

ところで、上述のような制御プログラム（組込みソフトウェア）を開発する際には、従来、制御すべき機構の試作品（実機）が完成している必要がある。即ち、試作が完了して初めてメカを具体的に動かすことができ、それを使って組込みソフトウェアの開発を開始することができるわけである。

この組込みソフトウェアの開発は、試作品の完成後、その試作品を実際に動作させながら、以下の手順で行なわれる。つまり、まず、組込みソフトウェアの概略設計を行ない、その概略設計に基づいて詳細設計を行なってから、詳細設計の結果をコーディングして組込みソフトウェアを作成し、その組込みソフトウェアのデバッグを行なう。

【０００５】

【発明が解決しようとする課題】

しかしながら、上述のように、組込みソフトウェア（制御プログラム）の開発を行なう際には制御対象の機構の試作品（実機）が必要であるため、試作品が完成するまでは組込みソフトウェアの開発を開始することができず、その開発に時

間やコストがかかり非効率的であるという課題があった。

【０００６】

また、試作品が完成して組込みソフトウェアの開発を開始したとしても、機構の試作段階では評価結果に応じて試作品の設計変更を行なうことは度々あり、その都度、組込みソフトウェアの開発を中断して、設計変更後の試作品の完成を待ってから、設計変更後の試作品に応じた仕様変更や再コーディングを行なわなければならず、その開発にさらなる時間やコストがかかり極めて非効率的であるという課題もあった。

【０００７】

このように、従来、機構の設計（実機の作成）とその機構のための組込みソフトウェアの開発とのコンカレント性は全くなく、同時に実行することができなかったので、組込みソフトウェアの開発は極めて非効率的に行なわれている。このため、機構の設計と組込みソフトウェアの開発とをコンカレントに（並行して）遂行できるようにすることが望まれている。

【０００８】

本発明は、このような課題に鑑み創案されたもので、機構の設計とその機構を制御するための組込みソフトウェアの開発とをコンカレントに遂行できるようにして、その組込みソフトウェアの開発の効率化をはかり、開発期間の短縮，工数の削減を実現した、支援システムおよび支援システムを記録したコンピュータ読み取り可能な記録媒体を提供することを目的とする。

【０００９】

【課題を解決するための手段】

上記目的を達成するために、本発明の支援システム（請求項１）は、アクチュエータおよびセンサを含む複数の部品からなる機構を３次元的に設計する機構設計部と、該機構を３次元機構モデルとして内部に構築され該機構の動作をシミュレートする３次元機構モデルシミュレーション部と、該機構に組み込まれ該機構の動作を制御するための制御プログラムを組込みソフトウェアとして開発する組込みソフトウェア開発部と、該機構設計部による設計データを該３次元機構モデルに反映させるべく該機構設計部から該３次元機構モデルシミュレーション部へ

入力する第１インタフェース部と、該３次元機構モデルシミュレーション部の動作と該組込みソフトウェア開発部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該組込みソフトウェア開発部との間でデータ送受を行なう第２インタフェース部とをそなえて構成されていることを特徴としている。

【００１０】

このとき、該第１インタフェース部が、該３次元機構モデルシミュレーション部によるシミュレーション結果を、該機構の設計に反映させるべく該３次元機構モデルシミュレーション部から該機構設計部へ入力してもよい（請求項２）。

また、該組込みソフトウェア開発部が、該組込みソフトウェアの詳細設計を行なうべく該組込みソフトウェアの仕様を記述する状態遷移図・表を作成・編集する状態遷移図・表作成部を含み、該第２インタフェース部が、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該状態遷移図・表作成部との間でデータ送受を行なうように構成してもよい（請求項３）。

【００１１】

この場合、該状態遷移図・表作成部が、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行し、該第２インタフェース部が、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させてもよい（請求項４）。そして、該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させるように構成してもよい（請求項５）。

【００１２】

さらに、該組込みソフトウェア開発部が、開発中の該組込みソフトウェアを組み込まれたマイクロコンピュータチップを含み、該第２インタフェース部が、該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作との同期をとりながら該３次元機構モデルシミュレーション部と該マイクロ

コンピュータチップとの間でデータ送受を行なうように構成してもよい（請求項６）。

【００１３】

この場合、該マイクロコンピュータチップが、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行し、該第２インタフェース部が、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させてもよい（請求項７）。そして、該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させるように構成してもよい（請求項８）。

【００１４】

なお、該第２インタフェース部が、該組込みソフトウェア開発部から該３次元機構モデルシミュレーション部へ、該３次元機構モデルにおける該アクチュエータに対するアクチュエータ指令信号を受け渡すとともに、該３次元機構モデルシミュレーション部から該組込みソフトウェア開発部へ、該アクチュエータ指令信号に応じたシミュレーション結果として得られる、該３次元機構モデルにおける該センサからのセンサ信号を受け渡すように構成してもよい（請求項９）。

【００１５】

また、該アクチュエータに対する該アクチュエータ指令信号と該３次元機構モデルシミュレーション部からの該センサ信号との時間変化をリアルタイムで解析して表示しうる解析部をそなえてもよい（請求項１０）。

一方、本発明の記録媒体（請求項１１）は、アクチュエータおよびセンサを含む複数の部品からなる機構を制御すべくその機構に制御プログラムとして組み込まれる組込みソフトウェアの開発を支援する機能をコンピュータにより実現するための、支援プログラムを記録したコンピュータ読み取り可能なものであって、該支援プログラムとして、①該機構を３次元的に設計する機構設計部として、該コンピュータを機能させる機構設計プログラムと、②該機構を３次元機構モデル

として内部に構築され該機構の動作をシミュレートする３次元機構モデルシミュレーション部として、該コンピュータを機能させる３次元機構モデルシミュレーションプログラムと、③該組込みソフトウェアを開発する組込みソフトウェア開発部として、該コンピュータを機能させる組込みソフトウェア開発プログラムと、④該機構設計部による設計データを、該３次元機構モデルに反映させるべく該機構設計部から該３次元機構モデルシミュレーション部へ入力する第１インタフェース部として、該コンピュータを機能させる第１インタフェースプログラムと、⑤該３次元機構モデルシミュレーション部の動作と該組込みソフトウェア開発部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該組込みソフトウェア開発部との間でデータ送受を行なう第２インタフェース部として、該コンピュータを機能させる第２インタフェースプログラムとを記録したことを特徴としている。

【００１６】

このとき、該第１インタフェースプログラムが、該３次元機構モデルシミュレーション部によるシミュレーション結果を、該機構の設計に反映させるべく該３次元機構モデルシミュレーション部から該機構設計部へ入力させるように構成されてもよい（請求項１２）。

また、該組込みソフトウェア開発プログラムが、該組込みソフトウェアの詳細設計を行なうべく該組込みソフトウェアの仕様を記述する状態遷移図・表を作成・編集する状態遷移図・表作成部として、該コンピュータを機能させる状態遷移図・表作成プログラムを含むものであり、該第２インタフェースプログラムが、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作との同期をとりながら該３次元機構モデルシミュレーション部と該状態遷移図・表作成部との間でデータ送受を行なわせるように構成されてもよい（請求項１３）。

【００１７】

この場合、該状態遷移図・表作成プログラムが、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを

該複数のタスクとは別に実行させるように構成され、該第２インタフェースプログラムが、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させてもよい（請求項１４）。そして、該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該状態遷移図・表作成部の動作とを同期させてもよい（請求項１５）。

【００１８】

さらに、該第２インタフェースプログラムが、該３次元機構モデルシミュレーション部の動作と、開発中の該組込みソフトウェアを組み込まれたマイクロコンピュータチップの動作との同期をとりながら該３次元機構モデルシミュレーション部と該マイクロコンピュータチップとの間でデータ送受を行なわせるように構成されてもよい（請求項１６）。

【００１９】

この場合、該マイクロコンピュータチップが、複数のタスクを並行して実行するマルチタスクを採用し、該３次元機構モデルシミュレーション部のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを該複数のタスクとは別に実行し、該第２インタフェースプログラムが、該同期タスクを用いて該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させてもよい（請求項１７）。そして、該同期タスクの優先度を最も高く設定し、該同期タスクにより該複数のタスクの開始・停止を制御して、該３次元機構モデルシミュレーション部の動作と該マイクロコンピュータチップの動作とを同期させてもよい（請求項１８）。

【００２０】

【発明の実施の形態】

以下、図面を参照して本発明の実施の形態を説明する。

図３は、本発明の一実施形態としての支援システムを構築しうるコンピュータシステムを示す外観図である。

この図３に示すように、コンピュータシステム１００は、図４を参照しながら後述するごとくＣＰＵ２２１，ＲＡＭ２２２，ハードディスク２１１等を内蔵し

た本体部１０１と、この本体部１０１からの指示により表示画面１０２ａで画面表示を行なうＣＲＴディスプレイ１０２と、このコンピュータシステム１００にユーザの指示や文字情報を入力するためのキーボード１０３と、表示画面１０２ａ上の任意の位置を指定することによりその位置に表示されているアイコン等に応じた指示を入力するマウス１０４とをそなえて構成されている。

【００２１】

本体部１０１は、さらに、外観上、フロッピィディスク２１２（図４参照）やＣＤ－ＲＯＭ２１０がそれぞれ取り出し自在に装填されるフロッピィディスク装填口１０１ａおよびＣＤ－ＲＯＭ装填口１０１ｂを有しており、その内部には、装填されたフロッピィディスク２１２やＣＤＲＯＭ２１０をドライブする、フロッピィディスクドライバ２２４やＣＤＲＯＭドライバ２２５（図４参照）を有している。

【００２２】

ここでは、ＣＤ－ＲＯＭ（コンピュータ読み取り可能な記録媒体）２１０に、本発明の一実施形態としての支援プログラム３００（図２参照）が記録されている。

このＣＤ－ＲＯＭ２１０がＣＤ－ＲＯＭ装填口１０１ｂから本体部１０１内に装填され、ＣＤ－ＲＯＭドライバ２２５により、そのＣＤ－ＲＯＭ２１０に記録された支援プログラム３００がこのコンピュータシステム１００のハードディスク２１１内にインストールされる。

【００２３】

また、コンピュータシステム１００のハードディスク２１１には、図１を参照しながら後述するメカ設計部（移送設計・３次元ＣＡＤシステム）２０により設計されて定義された、設計対象機構についての設計データが格納され、その設計データに基づき、コンピュータシステム１００（後述する３次元リアルタイムシミュレーション装置３０）において、その機構についての３次元機構モデルが構築されるようになっている。

【００２４】

さらに、コンピュータシステム１００のハードディスク２１１には、図１を参

照しながら後述する組込みソフトウェア開発部４０により開発される制御プログラム（組込みソフトウェア）が格納される。ハードディスク２１１には、開発を完了した組込みソフトウェアが格納されるほか、開発中の組込みソフトウェアも格納・保存される。

【００２５】

図４は、本発明の一実施形態としての支援システムを構築しうるコンピュータシステムのハードウェア構成を示すブロック図であり、この図４に示すように、本実施形態のコンピュータシステム１００においては、バスライン２２０を介して、中央演算処理装置（ＣＰＵ）２２１，ＲＡＭ２２２，ハードディスクコントローラ２２３，フロッピィディスクドライバ２２４，ＣＤ－ＲＯＭドライバ２２５，マウスコントローラ２２６，キーボードコントローラ２２７，ディスプレイコントローラ２２８，モデム２２９およびＲＯＭ２３０が相互に接続されている。

【００２６】

そして、フロッピィディスクドライバ２２４およびＣＤ－ＲＯＭドライバ２２５は、図３を参照して説明したように、それぞれ、フロッピィディスク２１２およびＣＤ－ＲＯＭ２１０が装填され、装填されたフロッピィディスク２１２およびＣＤ－ＲＯＭ２１０に対するアクセスを行なうものである。

また、ハードディスクコントローラ２２３は、ハードディスク２１１を制御するとともにＣＰＵ２２１からの指示に従ってハードディスク２１１に対するアクセスを行なうものであり、マウスコントローラ２２６は、マウス１０４とコンピュータシステム１００との間のインタフェースとして機能するものであり、キーボードコントローラ２２７は、キーボード１０３とコンピュータシステム１００との間のインタフェースとして機能するものであり、ディスプレイコントローラ２２８は、ＣＰＵ２２１からの制御信号に応じてＣＲＴディスプレイ１０２の表示状態を制御するものである。

【００２７】

さらに、モデム２２９は、図示しない通信回線に接続され、図３および図４に示したものと同様構成の、他のコンピュータシステムとの通信を行なうためのも

のである。

前述したように、ＣＤ－ＲＯＭ２１０には支援プログラム３００（図２参照）が記憶されており、ＣＤ－ＲＯＭドライバ２２５により、そのＣＤ－ＲＯＭ２１０からその支援プログラム３００が読み込まれ、バスライン２２０を経由し、ハードディスクコントローラ２２３によりハードディスク２１１内に格納される。実際の実行に当たっては、ハードディスク２１１内のプログラム３００は、ＲＡＭ２２２上にロードされ、ＣＰＵ２２１により実行される。

【００２８】

支援プログラム３００は、図１に示す組込み支援システム１０を実現するためのものであり、ＣＰＵ１０が、バスライン２２０を介してＲＡＭ２２２におけるプログラム３００を実行することにより、支援システム１０を成すメカ設計部２０，３次元リアルタイムシミュレーション装置３０，状態遷移図・表作成編集装置４１，統合開発環境４３およびインタフェース部５０～５２としての機能（その詳細については後述）が実現され、本実施形態の支援システム１０が実現されるようになっている。

【００２９】

この支援プログラム３００は、前述のごとくＣＤ－ＲＯＭ２１０に記録された形態で提供されるほか、その他、フレキシブルディスク（フロッピィディスク２１２）等の、コンピュータ読取可能な記録媒体に記録された形態で提供される。そして、コンピュータシステム１００はその記録媒体からプログラム３００を読み取って、前述したようにハードディスク２１１等の内部記憶装置または外部記憶装置に転送し格納して用いる。また、そのプログラム３００を、例えば磁気ディスク，光ディスク，光磁気ディスク等の記憶装置（記録媒体）に記録しておき、その記憶装置から通信回線およびモデム２２９を介してコンピュータシステム１００に提供するようにしてもよい。

【００３０】

なお、本実施形態における記録媒体としては、上述したフレキシブルディスク，ＣＤ－ＲＯＭ，磁気ディスク，光ディスク，光磁気ディスクのほか、ＩＣカード，ＲＯＭカートリッジ，磁気テープ，パンチカード，コンピュータの内部記憶

装置（ＲＡＭやＲＯＭなどのメモリ），外部記憶装置等や、バーコードなどの符号が印刷された印刷物等の、コンピュータ読取可能な種々の媒体を利用することができる。

【００３１】

さて、図４に示すコンピュータシステム１００により実現される、本実施形態の支援システム１０は、アクチュエータやセンサを有し３次元的な動作を行なう機構の設計・開発に際し、機構を制御すべくその機構に組み込まれる制御プログラム（組込みソフトウェア）の開発を支援するためのものである。

そして、上述のような機能を実現すべく、本実施形態の支援システム１０は、図１に示すように、メカ設計部２０，３次元リアルタイムシミュレーション装置３０，状態遷移図・表作成編集装置４１，統合開発環境装置４３およびインタフェース部５０～５２としての機能を有するほか、開発された組込みソフトウェアを実際に格納されて機構制御を行なうマイクロコンピュータチップ（以下、マイコンチップという）４２も有している。

【００３２】

なお、図１は、本発明の一実施形態としての支援システムの機能的構成を示すブロック図である。

本実施形態のシステム１０は、メカ設計と組込みソフトウェア開発とをコンカレントに遂行するためのもので、大きく分けると、メカ設計部２０，３次元リアルタイムシミュレーション装置３０および組込みソフトウェア開発部４０から構成されている。

【００３３】

ここで、メカ設計部（意匠設計・３次元ＣＡＤシステム，機構設計部）２０は、アクチュエータおよびセンサを含む複数の部品からなる機構を３次元的に設計するものである。このメカ設計部２０と３次元リアルタイムシミュレーション装置３０との連携の詳細については、図５および図６を参照しながら後述する。

３次元リアルタイムシミュレーション装置（３次元機構モデルシミュレーション部）３０は、メカ設計部２０により設計された機構を３次元機構モデルとして内部に構築され、その機構の動作をシミュレートするものである。この３次元リ

アルタイムシミュレーション装置３０の詳細については、図１７～図８２を参照しながら後述する。

【００３４】

組込みソフトウェア開発部４０は、メカ設計部２０による機構の設計と並行してその機構に組み込まれその機構の動作を制御するための制御プログラムを組込みソフトウェアとして開発するものである。

また、本実施形態のシステム１０においては、インタフェース部５０～５２がそなえられるほか、組込みソフトウェア開発部４０は、状態遷移図・表作成編集装置４１，マイコンチップ４２および統合開発環境装置４３をそなえて構成されている。

【００３５】

インタフェース部（第１インタフェース部）５０は、メカ設計部２０による設計データを、３次元リアルタイムシミュレーション装置３０にて構築される３次元機構モデルに動的に反映させるべく、メカ設計部２０から３次元リアルタイムシミュレーション装置３０へ入力する機能を果たすとともに、３次元リアルタイムシミュレーション装置３０によるシミュレーション結果を、機構の設計に動的に反映させるべく３次元リアルタイムシミュレーション装置３０からメカ設計部２０へ入力する機能も果たすように構成されている。

【００３６】

このインタフェース部５０の機能は、メカ設計部２０と３次元リアルタイムシミュレーション装置３０とのいずれか一方に含まれてもよいし、これらのメカ設計部２０と３次元リアルタイムシミュレーション装置３０とに分割されて含まれていてもよい。

インタフェース部（第２インタフェース部）５１，５２は、いずれも、３次元リアルタイムシミュレーション装置３０の動作と組込みソフトウェア開発部４０の動作との同期をとりながら、３次元リアルタイムシミュレーション装置３０と組込みソフトウェア開発部４０との間でデータ送受を行なうものである。

【００３７】

また、本実施形態のシステム１０における組込みソフトウェア開発部４０は、

状態遷移図・表作成編集装置（状態遷移図・表作成部）４１，マイコンチップ４２および統合開発環境装置４３を含んで構成されている。

ここで、状態遷移図・表作成編集装置４１は、組込みソフトウェアの詳細設計を行なうべく組込みソフトウェアの仕様を記述し、組込みソフトウェアについてのタスク制御フローを作成するものである。この状態遷移図・表作成編集装置４１の詳細、および、状態遷移図・表作成編集装置４１と３次元リアルタイムシミュレーション装置３０との連携の詳細については、図７～図１４を参照しながら後述する。

【００３８】

マイコンチップ４２は、開発中の組込みソフトウェアを組み込まれ、実際に機構に搭載されてその機構の動作を制御するものであり、統合開発環境装置４３は、状態遷移図・表作成編集装置４１からのタスク制御フローを、Ｃ，Ｃ＋＋，アセンブラコードに変換し（コーディング，コードジェネレーション）、ターゲットマイコン用の実行モジュール（即ち、マイコンチップ４２に書き込まれるべき組込みソフトウェア）を作成するためのものである。

【００３９】

そして、インタフェース部５１は、３次元リアルタイムシミュレーション装置３０の動作と状態遷移図・表作成編集装置４１の動作との同期をとりながら、３次元リアルタイムシミュレーション装置３０と状態遷移図・表作成編集装置４１との間でデータ送受を行なうものである。

本実施形態では、図１０～図１４を参照しながら後述するごとく、状態遷移図・表作成編集装置４１が、複数のタスクを並行して実行するマルチタスクを採用し、３次元リアルタイムシミュレーション装置３０のシミュレーション動作期間中、複数のタスクを停止させるように機能する同期タスクを複数のタスクとは別に実行し、インタフェース部５１が、同期タスクを用いて３次元リアルタイムシミュレーション装置３０の動作と状態遷移図・表作成編集装置４１の動作とを同期させている。

【００４０】

このとき、インタフェース部５１は、同期タスクの優先度を最も高く設定し、

この同期タスクにより複数のタスクの開始・停止を制御して、３次元リアルタイムシミュレーション装置３０の動作と状態遷移図・表作成編集装置４１の動作とを同期させている。

このインタフェース部５１の機能は、３次元リアルタイムシミュレーション装置３０と状態遷移図・表作成編集装置４１とのいずれか一方に含まれてもよいし、これらの３次元リアルタイムシミュレーション装置３０と状態遷移図・表作成編集装置４１とに分割されて含まれていてもよい。

【００４１】

また、インタフェース部５２は、３次元リアルタイムシミュレーション装置３０の動作とマイコンチップ４２の動作との同期をとりながら、３次元リアルタイムシミュレーション装置３０とマイコンチップ４２との間でデータ送受を行なうものである。

本実施形態では、マイコンチップ４２も、状態遷移図・表作成編集装置４１と同様、複数のタスクを並行して実行するマルチタスクを採用し、３次元リアルタイムシミュレーション装置３０のシミュレーション動作期間中、該複数のタスクを停止させるように機能する同期タスクを複数のタスクとは別に実行し、インタフェース部５２が、同期タスクを用いて３次元リアルタイムシミュレーション装置３０の動作とマイコンチップ４２の動作とを同期させている。

【００４２】

このとき、インタフェース部５２も、インタフェース部５１と同様、同期タスクの優先度を最も高く設定し、この同期タスクにより複数のタスクの開始・停止を制御して、３次元リアルタイムシミュレーション装置３０の動作とマイコンチップ４２の動作とを同期させている。

このインタフェース部５２の機能は、３次元リアルタイムシミュレーション装置３０とマイコンチップ４２とのいずれか一方に含まれてもよいし、これらの３次元リアルタイムシミュレーション装置３０とマイコンチップ４２とに分割されて含まれていてもよい。

【００４３】

なお、本実施形態のシステム１０では、インタフェース部５１，５２が、それ

ぞれ、状態遷移図・表作成編集装置４１およびマイコンチップ４２から３次元リアルタイムシミュレーション装置３０へ、３次元機構モデルにおけるアクチュエータに対するアクチュエータ指令信号を受け渡すとともに、３次元リアルタイムシミュレーション装置３０から状態遷移図・表作成編集装置４１およびマイコンチップ４２へ、そのアクチュエータ指令信号に応じたシミュレーション結果として得られる、３次元機構モデルにおけるセンサからのセンサ信号を受け渡すように構成されている。

【００４４】

また、３次元リアルタイムシミュレーション装置３０において、ロジックアナライザ（解析部；図１５参照）をそなえてもよい。このロジックアナライザは、アクチュエータに対するアクチュエータ指令信号と３次元リアルタイムシミュレーション装置３０で得られたセンサ信号との時間変化をリアルタイムで解析して表示し、組込みソフトウェア開発者がその時間変化を確認できるようにするためのものである。

【００４５】

さらに、新たに、２次元操作パネルを構築する手段と３次元リアルタイムシミュレーション装置３０とを結び付けるインタフェース手段を有し、３次元機構モデルの操作を上記２次元操作パネル上からインタラクティブにできるように構成してもよい。この場合、２次元操作パネル上には、実際にマイコンチップ４２上にそなえられるスイッチ類のイメージが表示されている。そして、組込みソフトウェア開発者は、２次元操作パネル上のスイッチ類のイメージをマウス等により操作することで、マイコンチップ４２上のスイッチ類を操作した場合のシミュレーションを、３次元リアルタイムシミュレーション装置３０に実行させることができるようになっている。

【００４６】

またさらに、本実施形態の３次元リアルタイムシミュレーション装置３０は、ギア，カム，溝，クラッチ，ローラ，ラチェット等の各種機構についての機構モデルを作成する手段を有するとともに、ＤＣモータ，ステッピングモータ，ＡＣモータ，マイクロスイッチ／光学スイッチ，エンコーダ，ポテンショメータ等の

各種モータ／センサについての機構モデルを作成する手段を有しており、これらの手段を用いて、各種機構や各種モータ／センサの任意の組み合わせからなる機構を３次元機構モデルとして表現することができるようになっている。

【００４７】

一方、図２は、本実施形態のコンピュータ読み取り可能な記録媒体３００に記録された、支援プログラム３１０の構成を示す図である。

この図２に示す記録媒体３００は、支援プログラム３１０を記録されたＣＤ－ＲＯＭ２１０や、そのプログラム３１０をインストールされた状態のハードディスク２１１や、そのプログラム３１０をダウンロードされたフロッピィディスク２１２，ＲＡＭ２２２あるいはＲＯＭ２３０等を代表的に示したものである。また、記録媒体３００は、通信回線等で接続された複数の記録媒体であって、これらの記録媒体に、プログラム３１０を成す複数のプログラム３１１～３１６を分割して記録したものも含んでいる。

【００４８】

このような記録媒体３００に記録された、本実施形態の支援プログラム３１０は、アクチュエータおよびセンサを含む複数の部品からなる機構を制御すべくその機構に制御プログラムとして組み込まれる組込みソフトウェアの開発を支援する機能（つまり図１に示すシステム１０）を、図３や図４に示すようなコンピュータシステム１００により実現するためのもので、機構設計プログラム３１１，３次元機構モデルシミュレーションプログラム３１２，組込みソフトウェア開発プログラム３１３，第１インタフェースプログラム３１４，第２インタフェースプログラム３１５および統合開発環境プログラム３１６を含んで構成されている。

【００４９】

ここで、機構設計プログラム３１１は、前述したメカ設計部２０としてコンピュータシステム１００を機能させるものであり、３次元機構モデルシミュレーションプログラム３１２は、前述した３次元リアルタイムシミュレーション装置３０としてコンピュータシステム１００を機能させるものである。

組込みソフトウェア開発プログラム３１３は、前述した組込みソフトウェア開

発部４０としてコンピュータシステム１００を機能させるもので、前述した状態遷移図・表編集装置４１としてコンピュータシステム１００を機能させる状態遷移図・表作成プログラム３２０を含んで構成されている。

【００５０】

第１インタフェースプログラム３１４は、前述したインタフェース部５０としてコンピュータシステム１００を機能させるものである。この第１インタフェースプログラム３１４は、機構設計プログラム３１１と３次元機構モデルシミュレーションプログラム３１２とのいずれか一方に含まれてもよいし、これらのプログラム３１１および３１２に分割されて含まれていてもよい。

【００５１】

なお、機構設計プログラム３１１（メカ設計部２０）は、コンピュータシステム１００内にローディングされた第１インタフェースプログラム３１４（インタフェース部５０）と結合され、このコンピュータシステム１００内において、３次元機構モデルシミュレーションプログラム３１２の動作つまり３次元機構モデルの動作に連携して実行される。

【００５２】

第２インタフェースプログラム３１５は、前述したインタフェース部５１および５２としてコンピュータシステム１００を機能させるものである。

この第２インタフェースプログラム３１５のうちインタフェース部５１にかかる部分については、３次元機構モデルシミュレーションプログラム３１２と状態遷移図・表作成プログラム３２０とのいずれか一方に含まれてもよいし、これらのプログラム３１２および３２０に分割されて含まれていてもよい。

【００５３】

また、第２インタフェースプログラム３１５のうちインタフェース部５２にかかる部分については、３次元機構モデルシミュレーションプログラム３１２とマイコンチップ４２に格納される制御プログラム（組込みソフトウェア）とのいずれか一方に含まれてもよいし、これらのプログラム３１２および制御プログラムに分割されて含まれていてもよい。

【００５４】

統合開発環境プログラム３１６は、前述した統合開発環境装置４３としてコンピュータシステム１００を機能させるものである。

なお、本実施形態では、前述した通り、状態遷移図・表作成プログラム３２０（状態遷移図・表作成編集装置４１）やマイコンチップ４２がマルチタスクを採用している。

【００５５】

そのため、第２インタフェースプログラム３１５は、状態遷移図・表作成プログラム３２０またはマイコンチップ４２で実行される複数のタスクを、優先度を最も高く設定した、一つの同期タスクによって代表させ、その同期タスクを用いて、３次元機構モデルシミュレーションプログラム３１２（３次元リアルタイムシミュレーション装置３０）がシミュレーション動作を行なっている間は、状態遷移図・表作成プログラム３２０またはマイコンチップ４２での複数のタスクを停止させ、これにより、３次元機構モデルシミュレーションプログラム３１２による動作と、状態遷移図・表作成プログラム３２０やマイコンチップ４２による動作とを同期させている。

【００５６】

このとき、第２インタフェースプログラム３１５（インタフェース部５１，５２）は、コンピュータシステム１００で実行される際、３次元機構モデルシミュレーションプログラム３１２と状態遷移図・表作成プログラム３２０またはマイコンチップ４２（制御プログラム）との間において、以下のような通信を行なうことにより、同期タスクを用いた同期化を行なっている。

【００５７】

即ち、３次元機構モデルシミュレーションプログラム３１２が、状態遷移図・表作成プログラム３２０または制御プログラムの実行時間（T_S）を第２インタフェースプログラム３１５に通知する。

第２インタフェースプログラム３１５（インタフェース部５１，５２）は、その通知を受けて、通知された実行時間分だけ状態遷移図・表作成プログラム３２０または制御プログラムの実行を許容し、状態遷移図・表作成プログラム３２０または制御プログラムがその実行時間分だけ実行されると、状態遷移図・表作成

プログラム３２０または制御プログラムの実行をその段階で中断させ、その実行時間分だけのプログラム実行が終了したことと、そのプログラム実行により得られたアクチュエータ指令信号とを３次元機構モデルシミュレーションプログラム３１２に通知する。

【００５８】

この終了通知を受けた３次元機構モデルシミュレーションプログラム３１２は、３次元機構モデルを、状態遷移図・表作成プログラム３２０または制御プログラムの、その実行時間分の制御に応じた動作量だけ動作させ、その動作量に応じたセンサ信号を得てから、状態遷移図・表作成プログラム３２０または制御プログラムの次の実行時間を決定し、その決定した実行時間と前記センサ信号とを第２インタフェースプログラム３１５に通知する。

【００５９】

このような通信処理を繰り返すことにより、相互に同期がとられた状態で、３次元機構モデルシミュレーションプログラム３１２と状態遷移図・表作成プログラム３２０または制御プログラムとが実行され、３次元リアルタイムシミュレーション装置３０においては、その状態遷移図・表作成プログラム３２０または制御プログラムの実行による制御に従って３次元機構モデルが動作する。

【００６０】

また、コンピュータシステム１００（リアルタイムシミュレーション装置３０）内に構築された３次元機構モデルは、ＣＤ－ＲＯＭ２１０からこのコンピュータシステム１００内にローディングされた支援プログラム３１０を構成する３次元機構モデルシミュレーションプログラム３１２によりそのコンピュータシステム１００内で動作する。

【００６１】

そして、コンピュータシステム１００内に組み込まれた状態遷移図・表作成プログラム３２０や、マイコンチップ４２に組み込まれた制御プログラム（組込みソフトウェア）は、コンピュータシステム１００内にローディングされた第２インタフェースプログラム３１５（インタフェース部５１，５２）と結合され、このコンピュータシステム１００内において、３次元機構モデルシミュレーション

プログラム３１２の動作つまり３次元機構モデルの動作と同期して実行される。

【００６２】

上述のごとく、本実施形態の支援システム１０は、１つの装置（ここではコンピュータシステム１００）内に、メカ設計部２０，３次元リアルタイムシミュレーション装置３０，状態遷移図・表作成編集装置４１，統合開発環境装置４３およびインタフェース部５０～５２の全てを組み込んだ形態となっている。

本発明の支援システムは、上述のような形態以外に、複数のコンピュータシステム１００をモデム２２９および通信回線（図示省略）により相互に通信可能に接続し、これらのコンピュータシステム１００に、メカ設計部２０，３次元リアルタイムシミュレーション装置３０，状態遷移図・表作成編集装置４１，統合開発環境装置４３およびインタフェース部５０～５２としての機能を分担させた形態として構成することもできる。

【００６３】

この場合、機構設計プログラム３１１，３次元機構モデルシミュレーションプログラム３１２，状態遷移図・表作成プログラム３２０，第１インタフェースプログラム３１４，第２インタフェースプログラム３１５および統合開発環境プログラム３１６は、それぞれ、メカ設計部２０，３次元リアルタイムシミュレーション装置３０，状態遷移図・表作成装置４１，インタフェース部５０～５２および統合開発環境装置４３としての機能を果たすコンピュータシステム１００にインストールされることになる。

【００６４】

なお、本実施形態のシステム１０において、開発中の制御プログラム（組込みソフトウェア）をローディングしたマイコンチップ４２は、コンピュータシステム１００にＩ／Ｏボード（図示省略）を介して接続されており、制御プログラムに基づいてマイコンチップ４２から出力されたアクチュエータ指令信号（本来はモータドライバ等に伝達される電気信号）は、前記Ｉ／Ｏボードを介してコンピュータシステム１００に入力され、インタフェース部５２を介して３次元リアルタイムシミュレーション装置３０に入力される。そして、３次元リアルタイムシミュレーション装置３０は、そのアクチュエータ指令信号に応じて３次元機構モ

デルを動作させ、本来センサから出力される信号と同様のセンサ信号を生成し、そのセンサ信号を、インタフェース部５２および前記Ｉ／Ｏボードを介してマイコンチップ４２に入力している。

【００６５】

このようにして、本実施形態の支援システム１０では、３次元リアルタイムシミュレーション装置３０を中核に据えて、メカ試作品を作らなくても組込みソフトウェア（制御プログラム）の開発をメカ設計とは別個（コンカレント）に進めることができるようになっているほか、状態遷移図・表によるタスク制御記述法と３次元のリアルタイムシミュレーション装置３０とを組み合わせ、組込みソフトウェアをメカ試作品なしに効率よく開発することができるようになっている。

【００６６】

上述のごとく、本実施形態の支援システム１０は、リアルタイムに機構シミュレーションを行ないながらメカ設計を行なうメカ設計部２０と、機構をリアルタイムにシミュレーションする３次元リアルタイム・シミュレーション装置３０と、３次元リアルタイムシミュレーション装置３０上の仮想的な３次元機構モデルを実際のメカに見立てて組込みソフトウェアを開発する組込みソフトウェア開発部４０とをそなえ、これらをインタフェース部５０～５２を介して連携させるように構成されている。

【００６７】

このような本実施形態のシステム１０においては、メカ設計とそのメカを制御するための組込みソフトウェアの開発とがコンカレントに実行される。次に、その基本的な実行手順について簡単に説明する。

メカ設計者は、メカ設計部（意匠設計・ＣＧシステム，３次元ＣＡＤシステム）２０と３次元リアルタイムシミュレーション装置３０とを、インタフェース部５０を介して連動させ、関節，ギア，カム等の各種機構の動きを詳細に検証しながら、単体部品の設計、および、それらを組み立てるためのアセンブリ設計を進めていく。この設計過程で、メカ設計者は、設計変更が生じる度に、その変更部分についての設計データを、インタフェース部５０を介して３次元リアルタイムシミュレーション装置３０に動的に反映し、その時の全体アセンブリの動きを確

かめる。３次元リアルタイムシミュレーション装置３０で扱う３次元機構モデルはソフトウェアシステムであるため、実際に試作品を作製する場合と違って任意の設計変更を速やかに反映することが可能である。

【００６８】

一方、組込みソフトウェア開発者は、３次元リアルタイムシミュレーション装置３０を実機と見立て、この３次元リアルタイムシミュレーション装置３０と状態遷移図・表作成編集装置４１およびマイコンチップ４２とを、インタフェース部５１および５２を介して連動させながら組込みソフトウェア開発を進める。このとき、メカ設計側で発生した任意の設計変更は、３次元リアルタイムシミュレーション装置３０に速やかに反映されるため、組込みソフトウェアの開発がメカ側の設計変更によって滞ることはない。

【００６９】

組込みソフトウェア開発者は、まず、状態遷移図・表編集装置４１と３次元リアルタイムシミュレーション装置３０とを連動させながら組込みソフトウェアのタスク制御の詳細仕様（タスク制御フロー）を決定し、その結果を統合開発環境装置（マイコンシステム開発環境）４３に渡す。

統合開発環境装置４３では、状態遷移図・表作成編集装置４１から受け取ったタスク制御フローがＣ，Ｃ＋＋，アセンブラコードに変換され、ターゲットマイコン用の実行モジュール（組込みソフトウェア）が作成されてマイコンチップ４２に組み込まれる。

【００７０】

マイコンチップ４２と３次元リアルタイムシミュレーション装置３０とは、インタフェース部５２を介して連動することができるため、組込みソフトウェア開発部４０では、マイコンチップ４２に組み込んだ実行モジュールが仕様通りのタスク制御を行なうか否かを、３次元リアルタイムシミュレーション装置３０上の仮想メカモデル（３次元機構モデル）を動かすことにより確かめることができる。

【００７１】

このように、本実施形態のシステム１０では、３次元リアルタイムシミュレー

ション装置３０を仲介としてメカ設計および組込みソフトウェア開発をコンカレントに進め、最終的には、マイコンチップ４２と実機とを接続して、実機を用いた最終結合テストを行なうことで、極めて効率よく、メカ設計および組込みソフトウェア開発が可能になる。

【００７２】

次に、図５～図８２を参照しながら、本実施形態の支援システム１０の各部の詳細について説明する。

まず、メカ設計部（意匠設計・３次元CADシステム）２０と３次元リアルタイムシミュレーション装置３０との連携部分（インタフェース部５０）について、図５および図６を参照しながら説明する。

【００７３】

なお、図５はシステム１０の要部（メカ設計部２０および３次元リアルタイムシミュレーション装置３０）を示すブロック図、図６（Ａ）および図６（Ｂ）は、それぞれ、メカ設計部２０から３次元リアルタイムシミュレーション装置３０へ受け渡される図形データファイルおよびアセンブリデータファイルの具体例を示す図である。

【００７４】

図５に示すように、メカ設計部（意匠設計・３次元ＣＡＤシステム）２０からは、設計対象の各部品要素の図形データファイル（.slpファイルフォーマット）と、各部品要素を組み立てた状態にしたアセンブリデータファイル（.asyファイルフォーマット）とが出力され、これらのデータファイルは、３次元リアルタイムシミュレーション装置３０に読み込まれる。

【００７５】

このとき、メカ設計部２０と３次元リアルタイムシミュレーション装置３０との間には、前述したインタフェース部５０（図１参照）が介在しているが、図５に示す例では、インタフェース部５０としての機能が、メカ設計部２０と３次元リアルタイムシミュレーション装置３０とのいずれか一方に含まれているか、もしくは、これらのメカ設計部２０と３次元リアルタイムシミュレーション装置３０とに分割されて含まれている。従って、図５では、インタフェース部５０は図

示されていない。

【００７６】

図６（Ａ）に示すように、図形データファイルは.slpのファイルフォーマットであり、この.slpファイルフォーマットは、三角形ポリゴンの集合体を記述したもので、各三角形ポリゴンの各頂点の法線と３次元位置座標とを順次書き出したものである。また、図６（Ｂ）に示すように、アセンブリデータファイルは.asyのファイルフォーマットであり、この.asyファイルフォーマットでは、各部品要素の相互の配置情報が定義されている。

【００７７】

３次元リアルタイムシミュレーション装置３０は、メカ設計部２０から、.asyおよび.slpのファイルを読み込んだ後、関節，カム，ギア等の機構情報やモータ，センサの各特性情報を、３次元リアルタイムシミュレーション装置２０のＧＵＩ（Graphical User Interface）を通じて入力し、仮想メカモデル（３次元機構モデル）を完成させる。

【００７８】

次に、３次元リアルタイムシミュレーション装置３０と状態遷移図・表作成編集装置４１との連携部分（インタフェース部５１）について、図７～図１４を参照しながら説明する。

なお、図７はシステム１０の要部（３次元リアルタイムシミュレーション装置３０，状態遷移図・表作成編集装置４１およびインタフェース部５１）を示すブロック図、図８は状態遷移図の表示例を示す図、図９は状態遷移表の表示例を示す図、図１０（Ａ）～図１０（Ｃ）はマルチタスクについて説明するためのタイムチャート、図１１（Ａ）および図１１（Ｂ）は優先度によるタスクのスケジューリングについて説明するための図、図１２（Ａ）～図１２（Ｄ）は同期タスクを用いた同期化処理について説明するためのタイムチャートである。

【００７９】

図７に示すように、状態遷移図・表作成編集装置４１には、状態遷移で記述したタスク制御を実行するタスク制御部４１１と、キーボードやマウス等を使ってタスク制御部４１１によるタスク制御を編集するユーザインタフェース部４１２

と、タスク制御部４１１によるタスク制御の内容を表示する状態遷移表示部４１３と、３次元リアルタイムシミュレーション装置３０とのインタフェースを取るインタフェース部４１４とがそなえられている。状態遷移表示部４１３では、タスク制御部４１１によるタスク制御の内容（タスク制御フロー）が、図８に示すようにグラフィカルに状態遷移図の形で、または、図９に示すようにテーブル形式（状態遷移表の形）で表示されるようになっている。

【００８０】

また、３次元リアルタイム・シミュレーション装置３０には、リアルタイムに３次元機構モデルのシミュレーションを行なうシミュレーション部３１と、状態遷移図・表作成編集装置４１とのインタフェースを取るインタフェース部３２とがそなえられている。

インタフェース部４１４とインタフェース部３２とが、上述したインタフェース部５１を構成している。つまり、図７に示す例では、インタフェース部５１としての機能が、３次元リアルタイムシミュレーション装置３０と状態遷移図・表作成編集装置４１とに分割されて含まれている。

【００８１】

ところで、ファームウェア（状態遷移図・表作成編集部４１やマイコンチップ４２）と連携した３次元リアルタイムシミュレーション装置３０では、ファームウェア上での実時間と同じ時間スケールで、３次元機構モデル（仮想メカモデル）の動きなどをシミュレートする必要がある。

例えば図１０（Ａ）および図１０（Ｂ）に示すようにファームウェアのタスク１，２でT_S時間の計算処理が実行された場合には、引き続き、図１０（Ｃ）に示すように、３次元リアルタイムシミュレーション装置３０で、仮想メカ（３次元機構モデル）についてT_S時間分のシミュレーションを実行する。

【００８２】

ただし、図１０（Ａ）および図１０（Ｂ）に示すように、複数（図中、２つ）のタスクが並列に走るとき、複数のタスク間で同期をとりながら同時に処理の停止／開始を行なうことは、どのタスクが時間の管理を行なうか、どのように処理を中断／再開するかなどの問題がある。

そこで、本実施形態では、図１２（Ｃ）に示すような同期用のタスク（同期タスク）を用意し、この同期タスクに時間の管理と他のタスクの開始／停止の制御を行なわせることで、タスク内に特別な処理を追加せずに、ファームウェアと３次元リアルタイムシミュレーション装置３０とを同期させている。

【００８３】

このとき、リアルタイムＯＳの基本機能である、優先度によるタスクのスケジューリング機能を用いる。複雑な組込みソフトウェア（ファームウェア）では、リアルタイムＯＳを用いて、複数のタスクにより様々な処理を行なっている。リアルタイムＯＳでは、複数のタスクを並列に走らせることができるが、リソース（ＣＰＵ）は１つしかないので、ＯＳが、どのタスクにＣＰＵを割り当てるかスケジューリングを行なっている。

【００８４】

このスケジューリングを行なうべく、タスクに優先度を設定し、複数のタスクからＣＰＵ使用の要求が同時にきたとき、優先度の高いタスクからＣＰＵを割り当てていく手法が一般的に採用されている。例えば図１１（Ａ）および図１１（Ｂ）に示すように、優先度の低いタスクＬの処理中に優先度の高いタスクＨが処理を開始する時には、ＯＳは、タスクＬの処理を途中で停止させ、処理の実行権をタスクＨに移す。そして、ＯＳは、タスクＨの処理が終了すると、タスクＬの中断していた処理を再開させる。

【００８５】

このような優先度によるタスク切り替え機能を用い、同期タスクの優先度を他のタスクよりも高く設定すれば、同期タスクが他のタスクを開始／停止する制御を行なうことができる。このとき、通常のタスクには同期処理用の特別な処理は何も必要ない。

例えば図１２（Ａ）～図１２（Ｄ）に示すように、２つのタスク１，２を並列に実行する場合、新たに追加した同期タスクの優先度を最も高く設定し、タスク１，２でのT_S時間の計算処理に対応して３次元リアルタイムシミュレーション装置３０でT_S時間分のシミュレーションを実行している期間中は、ＣＰＵによる処理の実行権を同期タスクに移す。これにより、その期間中、タスク１，２の

処理が停止され、T_S時間分のシミュレーション終了後にタスク１，２の処理が再開される。

【００８６】

図１３は、本実施形態における同期タスクを用いた同期化処理について、より詳細に説明するためのブロック図であり、この図１３に示す例では、状態遷移図・表作成編集装置４１（マイコンチップ４２）と３次元リアルタイムシミュレーション装置３０とのインタフェース部５１（５２）を共有メモリ５３により実現している。

【００８７】

図１３に示す状態遷移図・表作成編集装置４１では、ｎ個のタスク１～ｎが、それぞれ内部メモリ４１５のデータを参照しながら並列的に実行される。同期タスクの処理中は、内部メモリ４１５のデータ（アクチュエータ指令信号）を読み出して３次元リアルタイムシミュレーション装置３０に送信し、この３次元リアルタイムシミュレーション装置３０からのデータ（センサ信号）を受信して内部メモリ４１５に書き込む。

【００８８】

このとき、各タスクの処理に合わせて個々のデータを３次元リアルタイムシミュレーション装置３０（仮想メカ）と送受信してもよいが、本実施形態では、通信を効率的に行なうために、内部メモリ４１５にバッファリングしておき、同期タスクが、必要なときにまとめてデータの送受信を行なう。通信手法としては、上述のような共有メモリ５３を用いる手法のほかに、Socket通信やメッセージ通信などによる方法を用いることもできる。

【００８９】

次に、同期タスクを用いた同期化処理について、図１４に示すフローチャート（ステップＳ１１～Ｓ１９）従って説明する。

この図１４に示すように、３次元リアルタイムシミュレーション装置３０からは前回のシミュレーション結果（センサ信号）とともに次回の計算時間（処理時間）T_Sが送信され、その時間T_Sが状態遷移図・表作成編集装置４１（マイコンチップ４２）に設定される（ステップＳ１１）。

【００９０】

そして、その時間T_Sだけ同期タスクの処理が停止される（ステップＳ１２）。つまり、その時間T_Sの間、複数のタスクの処理が実行され、各タスクの処理によって得られたアクチュエータ指令信号（モータ指令信号）が内部メモリ４１５に書き込まれる。本実施形態では、同期タスクは、通常は停止していて、ある時間T_Sが経過すると起動されるようにする。これは、ＯＳが持っているタイマ機能やスリープ機能により実現することができる。

【００９１】

時間T_Sだけ同期タスクの処理を停止して待機した後（ステップＳ１２）、同期タスクを起動して他の全てのタスクを停止し、同期タスクで、各タスクからの出力データ（指令信号）を内部メモリ４１５から読み込み（ステップＳ１３）、その出力データを共有メモリ５３に書き込んでから（ステップＳ１４）、共有メモリ５３に同期フラグ１（Loop Flag）として“０”を書き込み（ステップＳ１５）、３次元リアルタイムシミュレーション装置３０に処理の実行を許可する。これに伴い、共有メモリ５３において同期フラグ２（Ready Flag）が“０”から“１”に設定される。

【００９２】

この後、状態遷移図・表作成編集装置４１（マイコンチップ４２）では、３次元リアルタイムシミュレーション装置３０からのシミュレーション終了通知を待つとともに、他のタスクを停止したままにしておく。

３次元リアルタイムシミュレーション装置３０は、状態遷移図・表作成編集装置４１（マイコンチップ４２）からシミュレーション実行開始の指令を受け取ったら、T_S時間分のシミュレーションを行ない、T_S時間分のシミュレーションを完了すると、シミュレーション結果（センサ信号）と次回の計算時間（処理時間）T_Sとを共有メモリ５３に書き込むとともに、共有メモリ５３上の同期フラグ２（Ready Flag）を“０”に書き換え、状態遷移図・表作成編集装置４１（マイコンチップ４２）から次のシミュレーション実行開始の指令を受けるまで待機する。

【００９３】

そして、状態遷移図・表作成編集装置４１（マイコンチップ４２）では、共有メモリ５３上の同期フラグ２（Ready Flag）が“０”になったこと、つまり、シミュレーションが完了したことを認識すると（ステップＳ１６のＹＥＳルート）、共有メモリ５３上の同期フラグ１（Loop Flag）を“０”に書き換えるとともに（ステップＳ１７）、シミュレーション結果（センサ信号）である入力データを共有メモリ５３から読み込んでから（ステップＳ１８）、その入力データを内部メモリ４１５に書き込む（ステップＳ１９）。このとき、次回の計算時間（処理時間）T_Sも書き込まれ、再びステップＳ１１に戻って、上述と同様の処理を繰り返し実行する。

【００９４】

以上のような処理を繰り返すことにより、状態遷移図・表作成編集装置４１（マイコンチップ４２）においてマルチタスクを採用している場合であっても、状態遷移図・表作成編集装置４１（マイコンチップ４２）と３次元リアルタイムシミュレーション装置３０とを確実に同期させることができる。

なお、このとき、時間T_Sは固定でなく、シミュレーションの状況によって可変にしてもよい。このためには、前述したように、同期とともにシミュレーション時間T_Sを通信する。

【００９５】

また、状態遷移図・表作成編集装置４１（マイコンチップ４２）において、３次元リアルタイムシミュレーション装置３０へのデータ（指令信号等）は、３次元リアルタイムシミュレーション装置３０にシミュレーションを開始させる前に送信され、シミュレーション終了後、３次元リアルタイムシミュレーション装置３０からのデータ（センサ信号等）は、状態遷移図・表作成編集装置４１（マイコンチップ４２）側の計算を開始する前に受信され、内部メモリ４１５に書き出される。

【００９６】

さらに、３次元リアルタイムシミュレーション装置３０とマイコンチップ４２との間におけるインタフェース部５２も、図５～図１４により上述したインタフェース部５１と同様に構成されて、このインタフェース部５１と同様に機能し、

マイコンチップ４２においてマルチタスクを採用している場合であっても、マイコンチップ４２と３次元リアルタイムシミュレーション装置３０とを確実に同期させることができる。

【００９７】

また、状態遷移図・表作成編集装置４１の具体例としては、例えば、State Flow(MathWorks社製品)マニュアル，StateMate（iLogix社製品）マニュアル，ZIPC(CATS社製品)マニュアルなどに開示された技術が挙げられるほか、統合開発環境装置４３の具体例としては、Softune（富士通社製品）マニュアルなどに開示された技術が挙げられる。これらの状態遷移図・表作成編集装置４１と統合開発環境装置４３とは互いに連携しており、状態遷移図・表作成編集装置４１で作成したタスク制御フローは統合開発環境装置４３にロードされ、この統合開発環境装置４３により、最終的にターゲットマイコン用の実行モジュールがシームレスに作成されるようになっている。

【００９８】

そして、上述のように、メカ設計と組込みソフトウェアの開発とを、３次元リアルタイムシミュレーション装置３０を利用しながらコンカレントに進めた後は、実際のインタフェースの試作品を使って（実機とマイコンチップ４２とを接続して）、メカおよび組込みソフトウェアの結合試験（実機最終確認）を行なう。

またさらに、前述した通り、本実施形態の３次元リアルタイムシミュレーション装置３０にはロジックアナライザ（解析部）がそなえられている。このロジックアナライザにおいては、図１５に示すように、アクチュエータ（モータ）に対するアクチュエータ指令信号と３次元リアルタイムシミュレーション装置３０で得られたセンサ信号との時間変化が、同期アルゴリズム下で、リアルタイムで解析され、シミュレーション結果として表示される。これにより、組込みソフトウェア開発者がその時間変化を確認することができる。また、ロジックアナライザは、データを保存する機能や印刷する機能も有している。

【００９９】

図１６は、上述したロジックアナライザによる処理を含む、本実施形態でのシミュレーション処理全体の流れを説明するためのフローチャートであり、この図

１６に示すように、３次元リアルタイムシミュレーション装置３０によるシミュレーション処理中は、インタフェース部５１や５２による同期処理を行ないながら（ステップＳ２１）、状態遷移図・表作成編集装置４１やマイコンチップ４２からのアクチュエータ指令信号に基づくモータ処理を行なった後（ステップＳ２２）、そのアクチュエータ指令信号に応じたセンサ処理を行なう（ステップＳ２３）。これらのステップＳ２１～Ｓ２３の処理中は、Ｉ／Ｏボードやソケット通信による入出力が行なわれ、これらの処理に伴う出力値は、後から読み出すことができるようにバッファリングされる一方、入力値は、同期処理（ステップＳ２１）以外の部分で変化しないことが保証されているため、バッファリングは行なわれていない。

【０１００】

そして、ロジックアナライザによる処理が不必要である場合（ステップＳ２４のＮＯルート）、ステップＳ２１に戻る一方、必要である場合（ステップＳ２４のＹＥＳルート）、アクチュエータ指令信号やセンサ信号の記憶・表示が行われる（ステップＳ２５）。

このとき、本実施形態では、ループ毎に、シミュレーションすべき時間T_Sが変化することがあるため、ロジックアナライザの情報は、「時間」，「ＰＩＯ値」，「ＤＡ値」の３つの配列を用いて記憶される。

【０１０１】

なお、メモリ容量の節約の関係から、「時間」の配列としては、同期処理に際して３次元リアルタイムシミュレーション装置３０側から指定される制御ループ回数（図３３にて後述するｎ参照；時間T_Sに対応した回数）を記憶し、「ＰＩＯ値」の配列としては、各ビット毎に３２チャンネル分のデータを記憶し、「ＤＡ値」の配列としては、２チャンネル分のデータを１６ビットに量子化し３２ビットの整数として記憶している。

【０１０２】

ついで、本実施形態の支援システム１０における３次元リアルタイムシミュレーション装置３０について、図１７～図８２を参照しながら、より詳細に説明する。

図１７は、本実施形態の支援システム１０における３次元リアルタイムシミュレーション装置３０の動作概念を説明するためのの概念図であり、この図１７に示すように、３次元リアルタイムシミュレーション装置（「３Ｄモデルシミュレータ」と略記する）３０内には３次元機構モデル（「３Ｄモデル」と略記する）が構築されている。

【０１０３】

３Ｄモデルは、複数のリンク（製品を構成する個々の部品）の３次元形状とそれらのリンクの姿勢との関係を定義した３次元機構データによって定義される、製品の３次元モデルであり、リンクの姿勢を変更することによってその３Ｄモデルが動作することになる。

組込みソフトウェア開発部４０（状態遷移図・表作成編集装置４１やマイコンチップ４２）から３Ｄモデルシミュレータ３０には、モータ等のアクチュエータを動作させるためのアクチュエータ指令信号が送られ、３Ｄモデルシミュレータから組込みソフトウェア開発部４０には、センサのオン，オフ等をあらわすセンサ信号が送られる。また、組込みソフトウェア開発部４０と３Ｄモデルシミュレータ３０との間では同期信号が送受信される。なお、図１７において、同期信号の送受信機能を果たすインタフェース部５１，５２（図１参照）の図示は省略されている。

【０１０４】

図１８は、センサやモータの定義時の処理を説明するための、３Ｄモデルシミュレータ３０の模式図、図１９は、本実施形態の３Ｄモデルシミュレータ３０によるシミュレーション処理を説明するための模式図である。

３Ｄモデルは、３次元ＣＡＤ等のメカ設計部２０により構築され、その構築された３Ｄモデルに対し、ここでは、図１８に示すように、ユーザによりキーボード等の入力装置が操作されてモータやセンサが定義される。

【０１０５】

そのような定義が行なわれた後の実際のシミュレーションに当たっては、図１９に示すように、組込みソフトウェア開発部４０から送られてきたアクチュエータ指令信号（ここではモータ回転指令信号）に対応し、モータとして定義された

リンク（以下、「モータリンク」と称する）の姿勢を変化させ、さらに機構データに基づき、モータリンクに連結された他のリンクについて、モータリンクの姿勢変化に応じた量だけ動作させ、その結果として、センサとして定義されたリンク（以下、「センサリンク」と称する）の姿勢にも影響が出る。そこで、センサリンクの姿勢に応じセンサ信号を発生させ、そのセンサ信号を組込みソフトウェア開発部４０に送信する。

【０１０６】

図２０は、本実施形態の３Ｄモデルシミュレータ３０と組込みソフトウェア開発部４０（同期タスク）との動作タイミングを説明するための図であり、この図２０における図中のハッチング部分は、シミュレータ３０や組込みソフトウェア開発部４０がそれぞれ動作している期間を示している。

この図２０に示すように、シミュレータ３０と組込みソフトウェア開発部４０（同期タスクのオフ動作）とは、動作，停止のサイクルを交互に繰り返している。各サイクルにおいて、シミュレータ３０および組込みソフトウェア開発部４０は、次のような手順で動作する。ただし、シミュレータ３０と組込みソフトウェア開発部４０（同期タスク）との間の同期の詳細については後述する。

【０１０７】

（１）シミュレータ３０は、アクチュエータ指令信号を基に、シミュレーション時間T_S分のシミュレーションを行ない、センサ信号を出力する。

（２）シミュレータ３０は、現在の３Ｄモデルの干渉可能性から、次のループでのシミュレーション時間T_Sを決定し、その時間T_Sを組込みソフトウェア開発部４０に受け渡すとともに、動作開始指令（Ready Flag）を組込みソフトウェア開発部４０に受け渡す。

【０１０８】

（３）組込みソフトウェア開発部４０は、時間T_Sを受け取るとともに動作開始指令（Ready Flag）を受け取り、通り図１２～図１４を参照しながら前述した通り、同期タスクをオフとして、その時間T_S分の計算（制御）をマルチタスクで行なった後、シミュレータ３０側に動作終了信号（Loop Flag）を送り、次の動作開始指令（Ready Flag）を受け取るまでの間、同期タスクをオンとして動作

を停止する。

【０１０９】

（４）シミュレータ３０は、組込みソフトウェア開発部４０の動作終了信号（Loop Flag）を受け取って時間T_S分のシミュレーションを開始する。

シミュレータ３０と組込みソフトウェア開発部４０とは、以上の手順を繰り返すことにより、交互に同期しながら動作する。

図２１は、本実施形態の３Ｄモデルシミュレータ３０によるシミュレーション手順を説明するためのフローチャートである。

【０１１０】

この図２１に示すように、シミュレーションが開始されると、まず初期設定が行なわれて（ステップＳ１０１）、３Ｄモデルの各リンクの姿勢が初期化される。そして、シミュレーションを行なう時間間隔（シミュレーション間隔）T_Sが組込みソフトウェア開発部４０側に送信され（ステップＳ１０２）、組込みソフトウェア開発部４０側からシミュレーション間隔T_Sを受信した旨をあらわす信号を受け取ると（ステップＳ１０３のＹＥＳルート）、組込みソフトウェア開発部４０に計算開始命令を送信する（ステップＳ１０４）。

【０１１１】

組込みソフトウェア開発部４０によりシミュレーション間隔T_S分の計算が行なわれて、その終了が通知されると（ステップＳ１０５のＹＥＳルート）、ここでシミュレートしている３Ｄモデルを構成する全てのモータおよび全てのセンサを今回のT_S分の処理に関し未処理である旨を初期設定すれる（ステップＳ１０６）。

【０１１２】

次いで、ステップＳ２０１では、未処理のモータが存在しているか否か判定され、未処理のモータ１つずつについてステップＳ２０２～Ｓ２０７の処理が行なわれる。

すなわち、未処理のモータが存在している場合（ステップＳ２０１のＹＥＳルート）、組込みソフトウェア開発部４０側から送られてきた、今処理しようとしているモータに関するアクチュエータ指令信号に応じてそのモータの回転速度が

決定され（ステップＳ２０２）、その回転速度からT_S間に変位するモータの姿勢が決定される（ステップＳ２０３）。

【０１１３】

そして、機構関係定義によりそのモータに関連づけられた他の部品の姿勢が変更され（ステップＳ２０４）、部品間の最短距離ｄが測定され（ステップＳ２０５）、ステップＳ２０６，Ｓ２０７では、各モータの回転により測定される各最短距離ｄの中の最短距離Ｄが求められる。

ここでは、今までの最短距離Ｄよりも新たに測定された最短距離ｄが小さいか否かを判断し（ステップＳ２０６）、小さい場合（ＹＥＳルート）、新たに測定された最短距離ｄを最短距離Ｄに置き換えてから（ステップＳ２０７）、ステップＳ２０１に戻る一方、小さくない場合（ＮＯルート）、そのままステップＳ２０１に戻る。

【０１１４】

上述のようなステップＳ２０２～Ｓ２０７の処理は、ステップＳ２０１でＮＯルートをとるまで、つまり、未処理のモータが存在しないと判定されるまで、繰り返し実行される。

このようにして、全てのモータについてT_S間の姿勢変更が終了すると、姿勢変更後の３Ｄモデルが画面に表示される（ステップＳ２０８）。

【０１１５】

この後、今度は未処理のセンサが存在しているか否か判定され（ステップＳ３０１）、未処理センサが存在している場合（ＹＥＳルート）、未処理のセンサ１つずつについてセンサリンクの姿勢に応じたセンサ信号のオン，オフが検出される（ステップＳ３０２）。

全てのセンサについての処理が終了すると（ステップＳ３０１のＮＯルート）、組込みソフトウェア開発部４０に向けてそれら全てのセンサに関するセンサ信号が出力され（ステップＳ３０３）、さらに、部品間最短距離Ｄに応じたサンプリング間隔T_Sが決定され（ステップＳ４０１）、ステップＳ１０２に戻ってその決定された間隔T_Sが組込みソフトウェア開発部４０に送信される。

【０１１６】

シミュレータ３０では、以上の処理が繰り返し実行され、組込みソフトウェア開発部４０との同期をとりつつ、３Ｄモデルの動作シミュレーションが行なわれる。

図２２は、図２１に示すシミュレータの処理フローの一部分（図２１に一点鎖線で囲った部分）に代えて採用することのできる部分フローを示すフローチャートである。

【０１１７】

この図２２には、ステップＳ２０２とステップＳ２０３との間にステップＳ２１０が挿入されている。このステップＳ２１０では、今処理を行なっているモータの回転速度がゼロ（$\omega = 0$）か否かが判定され、今回の間隔T_Sの間にそのモータは回転しない（$\omega = 0$）のときは（ＹＥＳルート）、そのモータの動作に関しては部品間の距離ｄの測定は行なわずにステップＳ２１１に進んで$d = \infty$としてからステップＳ２０６へ移行し、このステップＳ２０６の判定でその距離$d = \infty$が距離Ｄを求める演算から除外されるようにしている。ステップＳ２１０で今処理を行なっているモータの回転速度がゼロではないと判定された場合（ＮＯルート）、ステップＳ２０３へ移行し、前述した通りの処理を行なう。

【０１１８】

部品間の距離ｄを求めるにはかなりの計算量を要するので、上述のごとく、モータが回転しないときはそのモータに関する距離ｄの測定を行なわないようにすることにより、シミュレータ３０での計算量を減らし、シミュレーションのスピードアップをはかることができる。

図２３は、モータとして定義されるリンク（モータリンク）の選び方を説明すべくモータリンクの一例を模式的に示す斜視図である。

【０１１９】

この図２３において、本来のモータは図中のリンクＡであるが、リンクＡ自体は土台に固定されているため、シミュレーション上は、その本来のモータによって一番最初に駆動されて３Ｄモデルで姿勢が変更されるリンクＢを、モータリンクとして定義する。モータリンクをこのように定義することにより、シミュレーションを容易化している。

【０１２０】

図２４（Ａ），（Ｂ）および図２５（Ａ），（Ｂ）は、センサとして定義されるリンク（センサリンク）の選び方を説明すべく、それぞれ、センサリンクの一例および他例を模式的に示す斜視図である。

これらの図２４（Ａ），（Ｂ）および図２５（Ａ），（Ｂ）において、本来のセンサはリンクＡであるが、ここでは、その本来のセンサによって検出されるリンクＢをセンサリンクとして定義する。そして、リンクＢの姿勢（位置）に応じて、図２４（Ａ）および図２５（Ａ）に示す状態を、センサがオフの状態として定義し、図２４（Ｂ）および図２５（Ｂ）に示す状態を、センサがオンの状態として定義する。このように、センサの場合も、モータの場合と同様、本来のセンサではなくそのセンサにより検出されるリンクをセンサとして定義し、そのセンサリンクの姿勢に応じてオン，オフを定義することにより、シミュレーションを容易化している。

【０１２１】

図２６は、モータの種類を説明するための図である。

この図２６に示すように、本実施形態では、モータを定義するにあたり「無段変速」と「有段変速」との二種類の定義方法が用意されている。

「無段変速」は、速度指令値と目標速度との間の比例定数を定義する定義方法である。この「無段変速」は、さらに、「正負に関係なく比例」させる方法と、二進数の「最上位ビットを符号」とみなす方法との２種類が用意されている。前者の方法では、例えば比例定数が１００のとき、二進数１０１（１０進数で－３）が－３００ｒｐｍ（マイナス符号は逆転方向への回転をあらわす）に変換される。また、後者の方法では、例えば二進数１０１について、最上位ビット‘1’とその他のビット‘０１’とに分け、最上位ビット‘1’はマイナス符号（‘０’はプラス符号）、残りの‘０１’を数値（１０進数で１）とし、それらを合わせた－１に比例定数１００を掛けて－１００ｒｐｍに変換する。

【０１２２】

「有段変速」は、速度指令値と目標速度との対応表が定義される。例えば図２６中の対応表では、二進数１０１に対し－２００ｒｐｍが定義されており、速度

指令値として二進数１０１が入力されると、対応表を参照し目標速度として－２００ｒｐｍが出力される。なお、対応表としては、図７５を参照しながら後述するようなものもある。

【０１２３】

本実施形態では、上述のような複数種類の定義方法を用意しておくことにより、モータの定義の自由度を向上させている。

図２７は、モータを一次遅れ系とみなしたときの、モータの回転速度ωの初期変化を示す図である。

ここでは、モータの回転速度変化を図２７に示すような一次遅れ系としてとらえて、モータの負荷による遅れやその影響を取り入れたシミュレーションを行なう。モータを一次遅れ系とみなすと、ユーザは、整定時間４Ｔを入力するだけでそのモータの特性を指定することができ、入力作業の繁雑さが低減される。

【０１２４】

以下に、モータを一次遅れ系とみなしたときの、シミュレーション間隔T_Sの間のモータの回転量Δθの算出式を導出する。なお、以下、シミュレーション間隔T_SをΔｔとして表記する。

時刻ｔにおけるモータ速度ωは、下式（１）により表される。

【０１２５】

【数１】

$$\omega = \omega_o + (\omega_i - \omega_o)\left(1 - e^{-\frac{t}{T}}\right) \qquad (1)$$

【０１２６】

ここで、ω_iは目標速度、ω_oは目標速度指令前のモータ速度、Ｔはモータの性質を表す整定時間４Ｔに比例した定数である。

上式（１）で表される時刻ｔにおけるモータ速度ωをω_{k-1}とすると、時刻ｔよりもΔｔ後の時刻ｔ＋Δｔのモータ速度ω_kは、下式（２）となる。

【０１２７】

【数２】

$$\omega_k=\omega_0+(\omega_1-\omega_0)\left(1-e^{-\frac{t+\Delta t}{T}}\right) \qquad (2)$$

【０１２８】

これらの（１），（２）式から下式（３）が得られる。

【０１２９】

【数３】

$$\omega_k=\omega_{k-1}+(\omega_1-\omega_{k-1})\left(1-e^{-\frac{\Delta t}{T}}\right) \qquad (3)$$

【０１３０】

Δｔの間にモータが回転する量Δθは、上式（３）をΔｔについて積分することにより、下式（４）として与えられる。

【０１３１】

【数４】

$$\Delta\theta=\int_0^{\Delta t}\left[\omega_{k-1}+(\omega_1-\omega_{k-1})\left(1-e^{-\frac{t}{T}}\right)\right]dt$$

$$=\omega_{k-1}\Delta t+(\omega_i-\omega_{k-1})\left\{\Delta t-T\left(1-e^{-\frac{\Delta t}{T}}\right)\right\} \qquad (4)$$

【０１３２】

モータを一次遅れ系とみなしたとき、上式（４）に基づいて、目標速度ω_i，シミュレーション時間Δｔ，そのシミュレーション時間Δｔだけ前の時刻におけるモータ速度ω_{k-1}および整定時間４Ｔによって、そのシミュレーション時間Δｔの間のモータの回転量Δθが求められる。

ただし、モータは、常に上記（４）式のように動作する必要はなく、例えばモータに指令される目標速度ω_iが変化してから整定時間４Ｔが経過した時点で速度をω_iに固定してもよい。

【０１３３】

図２８は、本実施形態のシミュレータ３０における、一次遅れを考慮したモータ駆動の処理を説明するためのフローチャートである。

この図２８に示すように、部品間距離に基づいてシミュレーション間隔Δｔ（＝T_S）が決定され（ステップＳ５０１）、組込みソフトウェア開発部４０からアクチュエータ指令信号が入力されると（ステップＳ５０２）、図２６に示すモータの定義に従って比例定数あるいは対応表が参照されて目標速度ω_iが決定される（ステップＳ５０３）。

【０１３４】

ステップＳ５０４では、目標速度ω_iが変更されたか否か（$\omega_i = \omega_{iold}$であるか否か）が判定され、今回の目標速度ω_iが変更されていた場合（ＮＯルート；$\omega_i \neq \omega_{iold}$）は、ステップＳ５０５に進み、目標速度$\omega_i$が変更された時点からの経過時間を示す$t_{total}$が０に初期化され、次回のステップＳ５０４での判定のためにω_{iold}にω_iが格納されてから（ステップＳ５０６）、ステップＳ５０７へ移行する。

【０１３５】

ステップＳ５０４で$\omega_i = \omega_{iold}$であると判定された場合（ＹＥＳルート）は、ステップＳ５０５，Ｓ５０６を経由することなく、ステップＳ５０７に進み、目標速度ω_iが変更された時点からの経過時間t_{total}が整定時間４Ｔに達したか否かが判定される。t_{total}が未だ４Ｔに達していない場合（ＮＯルート）は、ステップＳ５０８に進み、上式（３）に従ってモータの回転速度ωが求められ、さらにステップＳ５０９において、上式（４）に従ってモータの回転量Δθが求められる。

【０１３６】

一方、ステップＳ５０７において$t_{total} \geqq 4T$であると判定された場合（ＹＥＳルート）は、ステップＳ５１０に進み、モータの回転速度ωが目標速度ω_iと等しい速度に設定され、さらに、ステップＳ５１１において、モータの回転量Δθが、目標速度ω_iで時間Δｔだけ回転したときの回転量に設定される。

ステップＳ５０８，Ｓ５０９を経由した場合であっても、あるいはステップＳ

５１０，Ｓ５１１を経由した場合であっても、ステップＳ５１２においてt_{total}がΔtだけ増分され、ステップＳ５１３においてモータのトータルの回転量θが$\Delta\theta$だけ増分される。

【０１３７】

この図２８に示す処理が組込みソフトウェア開発部４０と同期をとりながら繰り返し実行されることにより、図２７に示す一次遅れ系とみなしたときの速度変化に従ってモータが回転する。

図２９，図３０および図３１（Ａ）～図３１（Ｃ）は、干渉チェックを用いたセンサの定義方法を説明するための図である。

【０１３８】

図２９では、接触センサＳ１の姿勢が複数のモータＭの姿勢の組合せにより決定される例が示されている。この場合、図２４（Ａ），（Ｂ）および図２５（Ａ），（Ｂ）を参照して説明した方法を採用すると、複数のモータＭそれぞれの姿勢の組合せでセンサＳ１のオン，オフを定義することになり極めて複雑な定義が必要となる。そこで、ここでは、センサＳ１と土台との干渉をチェックし、それらの間の干渉の有無によってセンサＳ１をオンあるいはオフとする。こうすることにより、ユーザに複雑な定義を要求することなく、センサＳ１のオン，オフを決定することができる。

【０１３９】

図３０では、スリットＳＬの移動により光量センサ（光電センサ）Ｓ２のオン，オフが多数回繰り返される例が示されており、この場合、図３１（Ａ）～図３１（Ｃ）に示すように光束を１つの仮想リンクと見なし、その仮想リンクとスリットＳＬとの干渉の有無によってセンサＳ２のオン，オフを判定する。

この場合、仮想リンクとの干渉をチェックする相手のリンクは、図３０に示すように相手のリンクが予め分かっている場合、そのリンクに限定してもよい。また、相手のリンクが不明である場合であっても、本来のセンサ自体をあらわすリンクは、その仮想リンクと常に干渉（接触）しているため、干渉をチェックすべき相手のリンクからは除外される。

【０１４０】

この仮想リンクによるセンサ出力決定方法は、ここに示した光電センサなどに限らず、例えばマイクロスイッチの接触部分を仮想リンクとして定義するなど、接触式の位置検出センサにも適用することができる。

図２４（Ａ），（Ｂ）および図２５（Ａ），（Ｂ）を参照して説明したセンサの定義方法は、ポテンショメータ，エンコーダ等の角度を検出するセンサなど、一自由度の変位を検出するセンサを表現するのに有効な方法である。しかし、ここに示したような、複数のモータの姿勢の組合せにより出力が決定されるように配置されたセンサ（図２９）など多自由度の組合せの結果として姿勢が決定されるセンサや、スリットカウンタ（図３０）などオン，オフの変化が多いセンサには、その定義方法を採用するとユーザに複雑な定義を要求することになる。そのため、ここに説明した、リンクどうしの干渉の有無によりセンサのオン，オフを判定する方法を採用することが好ましい。

【０１４１】

図３２は、シミュレータ３０と組込みソフトウェア開発部４０との間で同期をとるために送受信される信号を説明するためのタイミングチャート、図３３は、シミュレータ３０における３Ｄモデルの動作とソフトウェア開発部４０の動作との同期をとるための同期化手法を説明するためのフローチャート、図３４は、その同期を実現するための、シミュレータ３０側の処理を説明するためのフローチャートである。ここで、図３３で説明する同期化手法を実現するためのプログラムは、インタフェース部５１，５２を実現するための第２インタフェースプログラム３１５に組み込まれる。

【０１４２】

図３２に示す、ループフラグ（Loop Flag），レディフラグ（Ready Flag）およびタイムスケールは、それぞれ以下の意味をもつ信号である。

（１）Loop Flag

組込みソフトウェア開発部４０からシミュレータ３０へ受け渡す信号であって、組込みソフトウェア開発部４０が現在動作中であるか否かを表わしている。

【０１４３】

（２）Ready Flag

シミュレータ３０から組込みソフトウェア開発部４０へ受け渡す信号であって、シミュレータ３０による１つのシミュレーション間隔Δｔ（＝T_S）分のシミュレーションが終了したことを表している。

（３）タイムスケール

シミュレータ３０から組込みソフトウェア開発部４０へ受け渡す信号であって、シミュレータ３０が部品間最短距離に基づいて決定したシミュレーション間隔Δｔ（＝T_S）を表している。

【０１４４】

本実施形態では、シミュレーション間隔Δｔ（＝T_S）を指定するタイムスケールとしては、５ｍｓｅｃ，２ｓｅｃなどの具体的な時間を用いず、そのシミュレーション間隔Δｔ（＝T_S）の間に、図３３に示す処理ルーチンを通過する回数ｎを用いている。

図３２に示すように、ステップＳ６０１でｎ＝０か否かを判定し、ｎ＝０でない場合（ＮＯルート）には、ステップＳ６０２に進んでｎを１だけデクリメントし、本来の処理（遷移図・表作成作成プログラム３２０や制御プログラム）の実行に移っている。そして、ステップＳ６０１においてｎ＝０と判定された場合（ＹＥＳルート）のみ、Loop Flagを‘Ｌｏｗ’に設定し同期タスクを起動させてから（ステップＳ６０３）、ステップＳ６０４～Ｓ６０７の処理が実行される。

【０１４５】

ここでは、図３２～図３４を説明するに当たり、シミュレータ３０では、現在、シミュレーションが行なわれており、従って、Ready Flagは‘Ｌｏｗ’レベルであり、組込みソフトウェア開発部４０側では図３３のステップＳ６０４にとどまってReady Flagが‘Ｈｉｇｈ’に変化するを待っている段階にあるとする。このとき、シミュレータ３０における同期処理をあらわす図３４のルーチンでは、ステップＳ７０３のシミュレーションを行なっている途中にある。

【０１４６】

なお、シミュレータ３０側では、初期設定を行なってから（ステップＳ７０１）、ステップＳ７０２に進んで、Loop Flagが‘Ｌｏｗ’に変化するのを待ち、前述した図３２のステップＳ６０３の処理によりLoop Flagが‘Ｌｏｗ’に変化

すると、ステップＳ７０３に進み、シミュレーション間隔Δｔ分のシミュレーションが行なわれる。

【０１４７】

ここで、シミュレーションが終了すると、シミュレータは図３４のステップＳ７０４に移り、次のシミュレーション間隔（タイムスケール）Δｔが、上述した回数ｎにより決定され、ステップＳ７０５においてそのタイムスケールΔｔ（ｎ）が組込みソフトウェア開発部４０に送信され、さらにステップＳ７０６に進んでReady Flagが‘Ｈｉｇｈ’レベルに変更される。

【０１４８】

これに応じて、組込みソフトウェア開発部４０側では、図３３に示すステップＳ６０４を抜け出し（ＹＥＳルート）、ステップＳ６０５に進みタイムスケールｎが読み込まれ、さらにステップＳ６０６に進んでLoop Flagが‘Ｈｉｇｈ’に変更され同期タスクが停止される。その後、ステップＳ６０７において、シミュレータ３０側から送信されてきているReady Flagが‘Ｌｏｗ’に変化するのを待って本来の処理に進む。

【０１４９】

組込みソフトウェア開発部４０において遷移図・表作成作成プログラム３２０や制御プログラムは一定周期で繰り返し実行されており、前述したように、ステップＳ６０１ではｎ＝０か否かが判定され、ｎ≠０の場合（ＮＯルート）、ステップＳ６０２に進んでｎが１だけデクリメントされる。これを繰り返し、ステップＳ６０１でｎ＝０と判定されると（ＹＥＳルート）、ステップＳ６０３に進みLoop Flagが‘Ｌｏｗ’に変更されて同期タスクが起動され、その後、ステップＳ６０４において、シミュレータ３０側から送られてくるReady Flagが‘Ｈｉｇｈ’に変化するのを待つ状態となる。

【０１５０】

一方、シミュレータ３０側では、図３４のステップＳ７０６においてReady Flagが‘Ｈｉｇｈ’に変更された後、ステップＳ７０７において、組込みソフトウェア開発部４０側から送られてくるLoop Flagが‘Ｈｉｇｈ’に変化するのを待ち、前述した図３２のステップＳ６０６の処理によりLoop Flagが ‘Ｈｉｇｈ’に

変化したことを受けて、ステップＳ７０８に進み、Ready Flagを‘Ｌｏｗ’に変化させ、ステップＳ７０２に進んで、Loop Flagが‘Ｌｏｗ’に変化するのを待つ。前述したように、Loop Flagが‘Ｌｏｗ’に変化すると、ステップＳ７０３に進み、シミュレーション間隔Δｔ分のシミュレーションが行なわれる。

【０１５１】

以上の処理が繰り返されることにより、シミュレータ３０と組込みソフトウェア開発部４０（遷移図・表作成作成プログラム３２０や制御プログラム）は相互に同期をとられながら、それぞれの処理がマルチタスクにより実行される。

次に、リンク（部品）どうしの干渉可能性の評価方法について説明する。

図３５（Ａ）および図３５（Ｂ）は、干渉可能性の評価方法の説明図である。ここでは、リンクＡが速度ｖでリンクＢに向かって動いている例が示されている。

【０１５２】

図３５（Ａ）は、干渉する可能性のある２つのリンク（リンクＡとリンクＢ）が大きく離れている状態を示している。この場合、干渉可能性が小さいものと判定されて、シミュレーション間隔Δｔとして比較的大きな値を設定し、リンクＡが一回のシミュレーションで進む距離ｖΔｔを大きくすることにより、シミュレーションを高速に行なうことができる。

【０１５３】

また、図３５（Ｂ）は、リンクＡとリンクＢとが近づき、それらの間の距離が小さい状態を示している。この場合、干渉可能性が大きいものと判定されて、シミュレーション間隔Δｔとして比較的小さな値を設定し、リンクＡが一回のシミュレーションで進む距離ｖΔｔを小さくすることにより、高精度なシミュレーションを行なうことができる。

【０１５４】

このように、本実施形態では、部品間の距離に応じてシミュレーション間隔Δｔが変更され、高精度なシミュレーションと高速なシミュレーションとの両立、調和が図られている。

図３６（Ａ）～図３６（Ｃ）は、部品間の距離ｄとシミュレーション間隔Δｔ

との関係の各種の例を示す図である。

【０１５５】

これらの図３６（Ａ）～図３６（Ｃ）に示すいずれの例によっても、距離ｄが小さくてもシミュレーションがあまりにも低速化されないように、且つ、距離ｄが大きくてもシミュレーションがあまりにも高速化されないように、部品間の距離ｄに対するシミュレーション間隔Δｔが決定される。

次に、干渉可能性の評価のために部品をグループ分けする方法について説明する。

【０１５６】

図３７（Ａ）および図３７（Ｂ）は、部品のグループ分けを説明するための図である。

これらの図３７（Ａ）および図３７（Ｂ）においては、リンクl_0～リンクl_5の６つのリンクが示されており、モータリンクとして定義されたリンクl_3が動く場合を考える。

【０１５７】

この場合、モータリンクl_3の動きによって姿勢に影響を受けるリンクはリンクl_3～l_5であり、これらのリンクl_3～l_5は、モータリンクl_3の動きに対して１つの剛体と考えることができる。

また、モータリンクl_3の動きによっては姿勢に影響を受けないリンクl_0～l_2も同様に、これらを合わせたものを１つの剛体と考えることができる。

【０１５８】

そこで、モータリンクl_3を動作させる場合には、図３７（Ｂ）に示すように、そのモータリンクl_3を動かしたことにより姿勢に影響を受けるリンクのグループと影響を受けないリンクのグループとの２つのグループに分け、それら２つのグループそれぞれを剛体とみなし、それら２つの剛体の間の最短距離を求めることにより、その求めた距離を次のシミュレーション間隔Δｔを決定するための最短距離の候補の１つとすることができる。

【０１５９】

ただし、図３７（Ａ）および図３７（Ｂ）におけるリンクl_3とリンクl_2（例

えばモータの軸と歯車等）は、常に接触しているにも拘らず異なるグループに分けられてしまい、このままでは上記２つの剛体は常に干渉していることになってしまう。

そこで、本実施形態では、上記のようにして２つのグループに分けた後、一度、部品間距離を求め、既に接触・干渉しているリンクのうち、今注目しているモータリンクを動かしたときに動かない方のグループに属するリンク〔図３７（Ａ）および図３７（Ｂ）に示す例の場合リンク1_2〕をそのグループから除外した上で２つのグループそれぞれを剛体とみなし、それら２つの剛体の間の距離が求められる。

【０１６０】

図３８（Ａ）～図３８（Ｄ）は、１つの３Ｄモデル内に複数のモータリンクが定義されている場合のグループ分けを説明するための図である。

図３８（Ａ）は、３Ｄモデルをあらわしており、ここには、モータＭ１～モータＭ３の３つのモータリンクが定義されている。

このとき、図３７（Ａ）および図３７（Ｂ）を参照して説明したグループ分けを、モータＭ１のみを動かした時〔図３８（Ｂ）参照〕と、モータＭ２のみを動かした時〔図３８（Ｃ）参照〕と、モータＭ３のみを動かしたとき〔図３８（Ｄ）参照〕とのそれぞれについて行なう。

【０１６１】

そして、シミュレーション実行時には、対象としているモータリンクを動かしたときは、そのモータリンクに関しグループ分けしたグループ間の最短距離が求められ、それら複数のモータリンクの動きにより求められた複数の‘最短距離’の中からさらに一番最短の距離が求められる（図２１のステップＳ２０１～Ｓ２０７参照）。

【０１６２】

図３９は、グループ分けした部品の再グループ化を説明するための図である。

図３８（Ａ）～図３８（Ｄ）を参照して説明した手法によると、各モータリンクに関し、その３Ｄモデルを構成するほぼ全ての部品がどちらかのグループ（剛体）に属する結果となり、この場合、各剛体を定義するためのデータ量が膨大な

ものとなってしまい、極めて大きなメモリ容量を必要とする可能性がある。

【０１６３】

そこで、図３９に示すように、各モータリンクの動きによって動かない方のグループ〔図３９に示す例ではグループ（Ｄ），（Ｅ），（Ｆ）〕に属する部品群のうち、共通部分を抜き出して１つの剛体とするとともに、各グループ（Ｄ），（Ｅ），（Ｆ）からその共通部分を除いた部分をそれぞれ１つの剛体とする。そして、距離の測定の際には、共通部分からなる剛体とその共通部分を除いた部分の剛体とを合体させて、相手のグループ〔グループ（Ｄ），（Ｅ），（Ｆ）に対応してそれぞれグループ（Ａ），（Ｂ），（Ｃ）〕との間の距離を求めるようにしてもよい。これにより、共通部分を各グループに配置しておく場合〔図３８（Ａ）～図３８（Ｄ）の場合〕と比べ、これらのグループを定義するデータを格納しておくためのメモリの容量を減らすことができる。

【０１６４】

図４０は、複数のリンクを、モータリンクの動きによって姿勢に影響を受けるリンクのグループとそのモータリンクの動きによっては姿勢に影響を受けないグループとに分けるルーチンを説明するためのフローチャート（ステップＳ８０１～Ｓ８１１）、図４１は、図４０に示すのサブルーチンである検索ルーチンを説明するためのフローチャート（ステップＳ９０１～Ｓ９１１）である。

【０１６５】

ここでは、モータリンクの動きによって姿勢に影響を受けるリンクがリストＡにリストアップされ、それ以外のリンクがリストＢにリストアップされる。なお、リストＡおよびリストＢは、ここで説明するフローの実行に先立って白紙状態にクリアされているものとする。

図４０に示すように、まず、リンクを指し示すポインタＰによりモータリンク（現リンク）が指し示され（ステップＳ８０１）、そのポインタＰで指し示されるリンクがリストＡに追加される（ステップＳ８０２）。

【０１６６】

そして、ポインタＰで指し示されているリンクにチャイルド（ｃｈｉｌｄ）のリンクが存在しているか否かが判定される（ステップＳ８０３）。

ここで、チャイルドとは、例えば図３７（Ａ）および図３７（Ｂ）に示すリンク機構においてポインタがモータリンクl_3を指し示していたとき、そのモータリンクl_3が動くことによって一緒に動くリンクの１つ（ここではリンクl_4）をいう。この場合、リンクl_4が仮にモータリンクであったとしても今着目しているモータリンクはリンクl_3であり、リンクl_4はリンクl_3のチャイルドとなる。なお、ポインタＰが指し示すリンクと同格の複数のチャイルドが考えられるときであっても、チャイルドはそのうちの１つに限定され、チャイルドと考えられる他のリンクは、その１つの限定されたチャイルドのブラザ（ｂｒｏｔｈｅｒ）として定義される。

【０１６７】

ポインタＰが指し示すリンクにチャイルドがあったときは（ステップＳ８０３のＹＥＳルート）、ステップＳ８０４に進み、そのチャイルドのリンクを指し示すようにポインタＰを移し、図４１に示す検索ルーチンが呼び出される（ステップＳ８０５）。その検索ルーチンについては後述する。ステップＳ８０５での検索ルーチンの実行を終了すると、ステップＳ８０６に進む。なお、ステップＳ８０３において、ポインタＰが指し示すリンクにチャイルドが存在しないと判定された場合（ＮＯルート）は、直接、ステップＳ８０６に進む。

【０１６８】

ステップＳ８０６では、ポインタＰで指し示されたリンクが、ギアやカム等の受動部品を駆動する駆動リンクであるか否かが判定される。

ポインタＰが受動部品を駆動する駆動リンクであったときは（ＹＥＳルート）、ステップＳ８０７に進み、ポインタＰが指し示すリンクによって駆動される受動部品であるリンクを指し示すようにポインタＰが変更され、ステップＳ８０８において図４１の検索ルーチンが呼び出される。ステップＳ８０８での検索ルーチンの実行を終了すると、ステップＳ８０９に進む。なお、ステップＳ８０６において、ポインタＰが指し示すリンクが受動部品を駆動するリンクではないと判定された場合（ＮＯルート）、直接、ステップＳ８０９に進む。

【０１６９】

このステップＳ８０９まで進んだ段階では、リストＡが完成しており、このス

テップＳ８０９では、全リンクからリストＡにリストアップされたリンクを除いたリンクがリストＢに格納される。

次いで、リストＡにリストアップされた全てのリンクとリストＢにリストアップされた全てのリンクとの間で干渉チェックが行なわれ（ステップＳ８１０）、相互に干渉し合っていたリンクの組のうち、リストＢにリストアップされた方のリンクがリストＢから削除される（ステップＳ８１１）。

【０１７０】

図４１に示す検索ルーチンでは、まず、ステップＳ９０１において、ポインタＰが指し示すリンクがリストＡに既にリストアップされているか否かが判定され、既にリストアップされていた場合（ＹＥＳルート）は、そのままこの検索ルーチンを抜けてこの検索ルーチンが呼ばれた元のルーチンに戻る。一方、ステップＳ９０１において、ポインタＰが指し示すリンクがリストＡにリストアップされていないと判定された場合（ＮＯルート）、ステップＳ９０２に進み、ポインタＰが指し示すリンクがリストＡに追加される。

【０１７１】

そして、ステップＳ９０３でポインタＰが指し示すリンクにチャイルドが存在するか否かが判定され、チャイルドが存在していた場合（ＹＥＳルート）、ステップＳ９０４に進んでポインタＰがそのｃｈｉｌｄのリンクを指し示すように変更され、ステップＳ９０５において再度この図４１に示す検索ルーチンが呼び出される。このように、この検索ルーチンは何重にも重なって呼び出されることがあり、この検索ルーチンの実行を終了すると、その検索ルーチンを呼び出したステップ(例えばステップＳ９０５で呼ばれたときはそのステップＳ９０５)に戻る。ステップＳ９０５での検索ルーチンの実行を終了すると、ステップＳ９０６に進む。なお、ステップＳ９０３において、ポインタＰが指し示すリンクにチャイルドが存在しないと判定された場合（ＮＯルート）、直接、ステップＳ９０６に進む。

【０１７２】

ステップＳ９０６では、ポインタＰが指し示すリンクにブラザが存在するか否かが判定され、ブラザが存在していた場合（ＹＥＳルート）、ステップＳ９０７

に進んでポインタＰがそのブラザのリンクを指し示すように変更され、ステップＳ９０８において再度この検索ルーチンが呼び出される。ステップＳ９０８での検索ルーチンの実行を終了すると、ステップＳ９０９に進む。なお、ステップＳ９０６において、ポインタＰが指し示すリンクにブラザが存在しないと判定された場合（ＮＯルート）、直接、ステップＳ９０９に進む。

【０１７３】

ステップＳ９０９では、ポインタＰで指し示されるリンクが、ギアやカム等の受動部品を駆動する駆動リンクであるか否かが判定され、駆動リンクであった場合（ＹＥＳルート）、ステップＳ９１０に進んでポインタＰがその駆動リンクを指し示すように変更され、ステップＳ９１１においてこの検索ルーチンが再度呼び出される。ステップＳ９１１での検索ルーチンの実行を終了すると、検索ルーチンを呼び出したステップに戻る。なお、ステップＳ９０９において、ポインタＰが指し示すリンクが受動部品を駆動するリンクではないと判定された場合（ＮＯルート）、直接、検索ルーチンを呼び出したステップに戻る。

【０１７４】

図４２は、図４０および図４１に示すルーチンの動作説明用のリンク機構モデルを模式的に示す斜視図、図４３は、図４２に示すリンク機構モデルのデータ構造を説明するための図である。

図４２に示すように、リンクＡはモータ、リンクＢおよびリンクＣは、それぞれ、リンクＡ(モータ)の回転に伴って一緒に回転するシャフトおよびギア、リンクＤはリンクＣと噛合しているギアである。また、リンクＥはリンクＤ（ギア）の回転に伴って一緒に回転するシャフト、リンクＦはリンクＤ（ギア）と噛合しているギアである。

【０１７５】

データ構造上は、図４３に示すように、リンクＡとリンクＤとリンクＦとが同格に並び、リンクＢはリンクＡのチャイルド、リンクＣはリンクＢのブラザ、リンクＥはリンクＤのチャイルドの関係にある。

図４２および図４３に示すリンク機構モデルについて、図４０および図４１に示すルーチンの実行によりリストＡを作成することを考える。ここでは、リンク

Ａ（モータ）から検索を始める。すなわち、まず、図４０のステップＳ８０１においてリンクＡが指し示されるようにポインタＰが指定される。以下の〔〕内の大文字アルファベットは、リストＡにリストアップされるリンク
をあらわしている。

【０１７６】

（１）ポインタＰがリンクＡを指し示す状態でスタートする。〔Ａ〕

（２）ポインタＰがリンクＡからそのチャイルドであるリンクＢに変更され、検索ルーチンが実行される。〔Ａ，Ｂ〕

（３）ポインタＰがリンクＢからブラザであるリンクＣに変更され、検索ルーチンが実行される。〔Ａ，Ｂ，Ｃ〕

（４）ポインタＰがリンクＣの受動部品であるリンクＤに変更され、検索ルーチンが実行される。〔Ａ，Ｂ，Ｃ，Ｄ〕

（５）ポインタＰがリンクＤからそのチャイルドであるリンクＥに変更され、検索ルーチンが実行される。〔Ａ，Ｂ，Ｃ，Ｄ，Ｅ〕

（６）リタンでポインタＰがリンクＤに戻る。

【０１７７】

（７）ポインタＰがリンクＤからその受動部品であるリンクＦに変更され、検索ルーチンが実行される〔Ａ，Ｂ，Ｃ，Ｄ，Ｅ，Ｆ〕

（８）リタンでポインタＰがリンクＤに戻る。

（９）リタンでポインタＰがリンクＣに戻る。

（１０）リタンでポインタＰがリンクＢに戻る。

【０１７８】

（１１）リタンでＰがリンクＡに戻る。

（１２）終了

以上の手順を踏んで、図４２に示す全てのリンク（リンクＡ～リンクＦ）がリストＡにリストアップされる。

次に、１つのモータリンクの動きによって姿勢の変化を受けるグループ内での干渉可能性の評価方法について説明する。

【０１７９】

モータリンクの動きによって干渉する可能性があるのは、図３７（Ａ）および図３７（Ｂ）を参照して説明したような、そのモータリンクの動きによって姿勢に影響を受けるリンクのグループと姿勢に影響を受けないリンクのグループとの間だけでなく、図４２に示すように姿勢に影響を受けるグループの内部でも干渉が生じる可能性がある。

【０１８０】

そこで、そのグループの内部で干渉する可能性のあるリンクの組を発見するために、ここでは、シミュレーションに先立って、モータリンクを少しずつ動かしながら、そのモータリンクを動かすことによって姿勢に影響を受けるグループ内の全てのリンクどうしの間の干渉チェックを行なう。このような総当たり的な干渉チェックによって干渉したリンクの組は実際のシミュレーションにあたっても相互間の距離を求める対象に加え、シミュレーション間隔Δｔ（＝T_S）を決定するための最短距離の候補の１つとする。これにより、正確なシミュレーションを行なうことができる。

【０１８１】

ただし、モータリンクを動かしたときに姿勢に影響を受けるグループ内において、例えば図４２に示すモータや歯車のように、常に接触しているリンクの組も存在する。そこで、モータリンクを少しずつ動かして干渉チェックを行なった際に、そのモータリンクの全ての姿勢で常に干渉・接触していたリンクの組、例えば歯車等で関係づけられたリンクの組はリンク間の距離を求める対象から排除される。

【０１８２】

図４４（Ａ）～図４４（Ｄ）は、モータリンクの動きによって姿勢に影響を受けるリンクグループ内で、シミュレーション時に距離を求めるリンクの組を抽出する手法を具体的に説明するための図である。

図４４（Ａ）は、ある１つのモータの動きによって姿勢に変化を受けるグループを構成する全部品を示しており、ここでは一例として、図４４（Ｂ）に示すように、そのモータを－１８０°から１８０°まで３６°おきに回転させ、各回転ごとに干渉チェックを行なうこととする。この場合、モータの回転およびリンク

内の干渉チェックは１１回行なわれる。

【０１８３】

その結果、例えば図４４（Ｃ）に示すような、干渉したリンクの組とその干渉回数とが得られたものとする。１１回の干渉チェックを行なったところ、リンクＢとリンクＥとの干渉、および、リンクＤとリンクＥとの干渉は１１回発生（常に干渉・接触）しているため、図４４（Ｄ）に示すように、これら２つの組はシミュレーション時における距離測定の対象からは除外される。

【０１８４】

次に、本発明の支援システム１０を構成する３Ｄモデルシミュレータ３０の別の実施態様について、図４５～図６４を参照しながら説明する。

図４５は、本実施形態における別の実施態様の３Ｄモデルシミュレータ３０によるシミュレーション処理全体の処理の流れを説明するためのフローチャートであり、この図４５に示すように、ここでは、シミュレータ３０と組込みソフトウェア開発部４０との同期処理（ステップＳ１００１），モータ処理（ステップＳ１００２），関節不良動作処理（ステップＳ１００３），センサ処理（ステップＳ１００４）および組込みソフトウェア開発部４０の実行時間ΔＴ（＝T_S）の決定処理（ステップＳ１００５）が繰り返し実行される。

【０１８５】

シミュレータ３０と組込みソフトウェア開発部４０との同期処理（ステップＳ１００１）については、既に図１２～図１４，図２０や図３２～図３４により前述した通りであるため、その説明は省略する。それ以外の処理について、以下順次説明する。

図４６は、モータ処理ルーチンを説明するためのフローチャートである。この図４６に示すフローチャートは、基本的には、前述した実施形態における、図２１の一点鎖線で囲った部分からなるモータ処理にモータの故障の概念を取り入れたものである。

【０１８６】

図４６に示すように、ステップＳ１１０１では、最短距離Ｄを求めるために先ずその最短距離Ｄに∞が初期入力され、ステップＳ１１０２では、未処理のモー

タが存在しているか否かが判定され、未処理のモータが存在する場合（ＹＥＳルート）、未処理のモータ１つずつについてステップＳ１１０３～Ｓ１１１０の処理が行なわれる。

【０１８７】

即ち、まず、未処理のモータそれぞれについて、そのモータが故障しているか否かが判定される（ステップＳ１１０３）。モータの故障の有無は、ユーザによりあらかじめ設定され、フラグとして保存されている。

故障したモータについては処理を行なわず（ステップＳ１１０３のＹＥＳルートからステップＳ１１０２）、故障していないモータについては（ステップＳ１１０３のＮＯルート）、組込みソフトウェア開発部４０側から送られてきた、今処理しようとしているモータに関するアクチュエータ指令信号に応じて、そのモータの姿勢変位量Δθが決定される（ステップＳ１１０４）。

【０１８８】

そして、Δθ＝０であるか否かを判定し、Δθ＝０である場合（ＹＥＳルート）には、ステップＳ１１０２に戻る一方、Δθ＝０でなければ（ＮＯルート）、そのモータのそれまでの姿勢θに今回の変位量Δθが加算され（ステップＳ１１０６）、そのモータリンクの移動に伴って移動するリンクの姿勢変位量を計算するリンク姿勢移動サブルーチン（図４７を参照しながら後述）が実行される（ステップＳ１１０７）。

【０１８９】

その後、部品間最短距離ｄが求められ（ステップＳ１１０８）、ｄ＜Ｄか否かが判定され（ステップＳ１１０９）、ｄ＜Ｄでないとき（ＮＯルート）はステップＳ１１０２に戻る一方、ｄ＜Ｄのとき（ＹＥＳルート）、すなわち、今回求めた最短距離ｄがこれまでに求めた最短距離Ｄよりも短かいときは、最短距離Ｄを今回の求めた最短距離ｄに置き換えて、最短距離Ｄを更新する（ステップＳ１１１０）。

【０１９０】

そして、全てのモータに関し、今回の演算時間ΔＴの間の姿勢変更が終了すると、このモータ処理を終了する。

図４７は、図４６に示すモータ処理ルーチン（ステップＳ１１０７）で実行されるリンク姿勢移動サブルーチンを説明するためのフローチャートである。

この図４７に示すように、リンク姿勢移動サブルーチンでは、まず、リンク姿勢移動（図４６のステップＳ１１０７でコールされたときは、モータリンクの姿勢移動）を受けて（ステップＳ１２０１）、その姿勢移動のあったリンクに対する受動リンクに未計算の受動リンクが存在するか否か判定する（ステップＳ１２０２）。

【０１９１】

未計算の受動リンクが存在する場合（ＹＥＳルート）、今回姿勢移動のあったリンクとそのリンクに対する受動リンクとの間にクラッチが存在するか否かを判定し（ステップＳ１２０３）、クラッチが存在する場合（ＹＥＳルート）、そのクラッチ部品の姿勢がオンの位置にあるか否かを判定する（ステップＳ１２０４）。クラッチ部品の存在や、クラッチ部品の姿勢とクラッチのオン，オフとの関係は、ユーザにより予め定義される。

【０１９２】

クラッチ部品が存在しない場合（ステップＳ１２０３のＮＯルート）、あるいは、クラッチ部品が存在していてもオン状態にある場合（ステップＳ１２０４のＹＥＳルート）、ステップＳ１２０５に進み、ギア，カム関係に基づいて受動部品の移動量を計算し、さらにステップＳ１２０６において、今回移動量を計算した受動部品を能動部品としたときのその能動部品に対する受動部品の移動量を求めるために、再度、このリンク姿勢移動サブルーチンがコールされ、そのサブルーチンを実行してからステップＳ１２０２へ戻る。なお、ステップＳ１２０４でクラッチ部品がオン状態ではないと判定された場合（ＮＯルート）、直接、ステップＳ１２０２に戻る。

【０１９３】

このようにして、リンク姿勢移動サブルーチンは、図４１に示す検索ルーチンと同様、再帰的に実行され、この再帰的な実行により、１つのモータを動かしたときに他の部品の移動量が高速に求められる。

図４８は、図４５のステップＳ１００３で実行される関節不良動作処理ルーチ

ンを説明するためのフローチャートである。

【０１９４】

この図４８に示すように、関節不良動作不良ルーチンでは、まず、３次元機構モデルを画面表示した後（ステップＳ１３０１）、ユーザが新たに不良リンクを指定したか否か（ステップＳ１３０２）、未処理のユーザ指定不良リンクが存在するか否か（ステップＳ１３０５）、未処理の、干渉が起きたリンクが存在するか否か（ステップＳ１３１０）、および不良リンクが存在するか否か（ステップＳ１３１４）が、順次、判定される。

【０１９５】

ステップＳ１３０２において、ユーザが新たに不良リンクを指定したと判定されると（ＹＥＳルート）、ステップＳ１３０３に進み、その新たに指定された不良リンクを駆動しているモータを検索し、そのモータとそのモータの移動方向とをリストに追加してから（ステップＳ１３０４）、ステップＳ１３０５へ進む。

ステップＳ１３０５において未処理のユーザ指定不良リンクが存在すると判定されると（ＹＥＳルート）、ステップＳ１３０６に進み、その不良リンクを駆動しているモータの移動方向が逆転したか否かが判定され、逆転していない場合（ＮＯルート）はステップＳ１３０７に進んでそのモータの姿勢を旧姿勢に戻し、そのモータの姿勢が旧姿勢に戻されたことに対応してリンクの姿勢を旧姿勢に戻すために、図４７に示すリンク姿勢移動サブルーチンがコールされて実行された後（ステップＳ１３０８）、ステップＳ１３０５に戻る。一方、ステップＳ１３０６においてそのモータの移動方向が逆転したと判定されると（ＹＥＳルート）、ステップＳ１３０９に進み、そのモータを不良リンクのリストから削除してから、ステップＳ１３０５に戻る。

【０１９６】

これにより、不良リンクが、ある一方向に移動しようとしたときに何らかの原因で引っ掛かって動かなくなり、その不良リンクが一旦逆方向に移動することによりその引っ掛かりが解除されて正常に戻るという動作がシミュレートされる。

ステップＳ１３１０において、未処理の、干渉が起きたリンクが存在する旨判定されると（ＹＥＳルート）、ステップＳ１３１１に進み、その不良リンクを駆

動しているモータの検索が行なわれ、そのモータの姿勢が旧姿勢に戻され（ステップＳ１３１２）、そのモータによって駆動される他のリンクの姿勢を旧姿勢に戻すべく、図４７に示すリンク機構サブルーチンがコールされて実行された後（ステップＳ１３１３）、ステップＳ１３１０に戻る。

【０１９７】

これにより、あるリンクが、他のリンクと干渉してそれ以上動かすことのできない状態をシミュレートすることができる。

ステップＳ１３１４において不良リンクが存在すると判定されると（ＹＥＳルート）、その３次元機構モデルを画像上に再描画する（ステップＳ１３１５）。これにより、ステップＳ１３０１における描画と合わせて、３次元機構モデルの再描画結果をシミュレーション画面上で見ると、不良リンクを受動部品に持ったモータおよびそのモータにより駆動される全リンクの動きが、不自然な状態（例えば、カクカクして、いかにもひっかかった様子）で表示される。

【０１９８】

なお、ここに示した例では、モータの姿勢を元に戻すことによって関節不良の状態を実現したが、関節値の上限あるいは下限を一時的に現在値に設定することで関節不良の状態を実現することも可能である。

図４９は、図４５のステップＳ１００４で実行されるセンサ処理ルーチンを説明するためのフローチャートである。

【０１９９】

前述の実施形態では、センサはオン，オフの信号を出力するオン／オフセンサとしていたが、ここでは、センサとして、オン／オフセンサのほか、ポテンショメータやエンコーダも存在し、さらにセンサの不良も考慮されている。

図４９に示すように、まず、ステップＳ１４０１で、未処理のセンサの有無が判定され、未処理のセンサが存在すると（ＹＥＳルート）、ステップＳ１４０２に進み、その未処理のセンサがオン／オフセンサであるか否かが判定される。オン／オフセンサであった場合（ＹＥＳルート）、ステップＳ１４０３において、オン／オフセンサ処理が行なわれ、そのオン／オフセンサを処理した結果のセンサ値（オンあるいはオフ）が出力され、その後、ステップＳ１４０１に戻る。

【０２００】

ステップＳ１４０２において、その未処理のセンサがオン／オフセンサではないと判定されると（ＮＯルート）、ステップＳ１４０４に進み、そのセンサがポテンショメータであるか否かが判定され、ポテンショメータであると判定されると（ＹＥＳルート）、ステップＳ１４０５に進んでポテンショメータ前処理が行なわれた後、ステップＳ１４０６でエンコーダ処理が行なわれその結果としてのセンサ値を出力してから、ステップＳ１４０１に戻る。

【０２０１】

一方、ポテンショメータではないと判定された場合（ＮＯルート）、つまり、未処理のセンサがエンコーダであると判定された場合、直接、ステップＳ１４０６のエンコーダ処理に進み、その結果としてのセンサ値を出力してから、ステップＳ１４０１に戻る。

図５０は、図４９のステップＳ１４０３で実行されるオン／オフセンサ処理ルーチンを説明するためのフローチャートである。

【０２０２】

このオン／オフセンサ処理ルーチンでは、そのオン／オフセンサの故障もシミュレートされており、図５０に示すように、まず、ステップＳ１５０１でそのオン／オフセンサがオンの状態のまま故障しているか否かが判定され、ステップＳ１５０３でそのオン／オフセンサがオフの状態のまま故障しているかが判定される。

【０２０３】

オン状態のまま故障している場合（ステップＳ１５０１のＹＥＳルート）、出力予定値としてオンが設定され（ステップＳ１５０２）、オフ状態のまま故障している場合（ステップＳ１５０３のＹＥＳルート）、出力予定値としてオフが設定され（ステップＳ１５０４）、ステップＳ１５０８に進んでその出力予定値がセンサ値として出力される。そのオン／オフセンサがオン状態あるいはオフ状態で故障しているか否かはユーザにより予め設定される。

【０２０４】

そのオン／オフセンサが故障していない場合（ステップＳ１５０１，Ｓ１５０

３のＮＯルート）、ステップＳ１５０５に進み、そのセンサリンクの姿勢から出力予定値（オンあるいはオフ）が決定され（ステップＳ１５０５）、次いで、性能劣化による出力予定値変更処理（ステップＳ１５０６）、および、チャッタリングによる出力予定値変更処理（ステップＳ１５０７）を実行してから、ステップＳ１５０８へ進む。

【０２０５】

図５１は、図５０のオン／オフセンサ処理ルーチンのステップＳ１５０６で実行されるセンサ性能劣化処理ルーチンを説明するためのフローチャートである。このルーチンでは、オン／オフセンサが故障（出力がオンまたはオフに固定）までいかなくても、応答性能が劣化し、そのオン／オフセンサの出力をオンまたはオフに変化させる状態が生じた後、暫らくは前の状態を保ち、遅れてオンまたはオフに変化する状態がシミュレートされる。

【０２０６】

図５１に示すように、まず、ステップＳ１６０１で、旧出力予定値ｄと出力予定値とが等しいか否かが判定され、変化していた場合（ＮＯルート）、ステップＳ１６０２に進み出力予定値がオンかオフかが判定される。出力予定値がオンの場合（ＹＥＳルート）、ステップＳ１６０３に進み、センサ出力がオフからオンに変化するときの遅れ時間をT_dに設定してから、ステップＳ１６０５へ進む。一方、出力予定値がオフの場合（ステップＳ１６０２のＮＯルート）、ステップＳ１６０４に進み、センサ出力がオンからオフに変化するときの遅れ時間をT_dに設定してから、ステップＳ１６０５に進む。

【０２０７】

ステップＳ１６０１で旧出力予定値ｄと出力予定値とが等しいと判定された場合（ＹＥＳルート）、あるいは、ステップＳ１６０３，Ｓ１６０４の処理の完了後、ステップＳ１６０５でT_d＞０か否か判定され、T_d＞０の場合（ＹＥＳルート）、ステップＳ１６０６に進んでT_dから組込みソフトウェア開発部４０の実行時間ΔＴが減算され、ステップＳ１６０７において出力予定値を反転してから（出力予定値がオンのときはオフ、出力予定値がオフのときはオン）、ステップＳ１６０８に進む。

【０２０８】

ステップＳ１６０５でT_d＞０でｈないと判定された場合（ＮＯルート）、あるいは、ステップＳ１６０７の処理の完了後、ステップＳ１６０８では、旧出力予定値ｄに今回の出力予定値が格納される。

図４５に示すように、センサ処理Ｓ１００４には繰り返し実行され、図５１に示すセンサ性能劣化処理も繰り返し実行され、T_d＞０のときは、この処理が繰り返される度に、T_dからΔＴが減算されT_d≦０に達するとセンサが変化することになる。

【０２０９】

図５２は、図５０のオン／オフセンサ処理ルーチンのステップＳ１５０７で実行される、チャタリングによる出力予定値変更処理ルーチンを説明するためのフローチャートである。このルーチンではオン／オフセンサがオンからオフ、あるいはオフからオンに変化するときのチャタリングがシミュレートされる。

この図５２に示すように、まず、ステップＳ１７０１で、旧出力予定値ｃと出力予定値とが等しいか否かが判定され、変化していた場合（ＮＯルート）、ステップＳ１７０２に進み、出力予定値がオンかオフかが判定される。出力予定値がオンの場合（ＹＥＳルート）、ステップＳ１７０３に進み、センサ出力がオフからオンに変化するときのチャタリング時間をＴｃに設定してから、ステップＳ１７０５に進む。一方、出力設定値がオフの場合（ステップＳ１７０２のＮＯルート）、ステップＳ１７０４に進み、センサ出力がオンからオフに変化するときのチャタリング時間をT_cに設定してから、ステップＳ１７０５に進む。

【０２１０】

そして、ステップＳ１７０５では、旧出力予定値ｃに出力予定値が設定され、この処理を終了した後、もしくは、ステップＳ１７０１で旧出力予定値ｃと出力予定値とが等しいと判定された場合（ＹＥＳルート）、ステップＳ１７０６でT_c＞０か否かが判定される。

そして、T_c＞０の場合（ＹＥＳルート）、ステップＳ１７０７に進んでT_cから組込みソフトウェア開発部４０の実行時間ΔＴが減算され、ステップＳ１７０８において、０～９９の値をとる乱数が発生されてその発生した乱数が予め設定

された発生確率（０～９９のうちのいずれかの値）と比較される。その結果、乱数＞発生確率の場合（ＹＥＳルート）、出力予定値のオン，オフを反転してから（ステップＳ１７０９）、処理を終了する。一方、ステップＳ１７０８で乱数＞発生確率でないと判定された場合（ＮＯルート）や、ステップＳ１７０６でT_c＞０ではないと判定された場合（ＮＯルート）は、そのまま、処理を終了する。

【０２１１】

このようにして、T_c＞０の間、確率的にオン，オフが変化するチャッタリングがシミュレートされる。

次に、図４９のステップＳ１４０６で実行されるエンコーダ処理について説明する。

図５３は、エンコーダの変位（角度等）に対する出力波形の一例を示す図、図５４（Ａ）～（Ｅ）は、エンコーダの、各種の出力波形を示す図である。

【０２１２】

ここでは、図５３に示すように、エンコーダの出力波形が、波長Ｌの周期的な関数ｆ（θ）である場合について考える。ただし、関数ｆ（θ）は、図５４（Ａ）～図５４（Ｅ）に示すような各種の波形をとることができる。ここで、波長Ｌは関数ｆ（θ）の一周期に対応するもので、例えば、このエンコーダがロータリ式のエンコーダであって回転角度１．０度ごとに一周期の信号を出力するエンコーダであるときは、Ｌ＝１．０（度）となる。

【０２１３】

ここで、変位θを電気角$\phi = 2\pi\theta / L$に変換した場合、関数ｆ（θ）が関数ｇ（φ）として表記されるものとして、以下、エンコーダ機能の原理を説明する。

前回行なったシミュレーション時のエンコーダの電気角をϕ_0、今回、すなわち前回行なったシミュレーションから組込みソフトウェア開発部４０の演算時間ΔＴだけ進んだ時刻の電気角をφ、エンコーダの電気角速度をωとすると、前回から今回までの間のシミュレーションサイクルにおけるエンコーダの移動量はωΔＴとなり、

$$\phi = \phi_0 + \omega \Delta T$$

が成立する。

【０２１４】

しかし、エンコーダはパルス数を測定するため、今回のシミュレーションサイクルにおけるωΔＴが２πよりも大きい場合、単純にｇ（φ）を出力するわけにはいかず、所定のパルス数のパルスを出力した後にｇ（φ）を出力する必要がある。

図５５は、エンコーダの出力処理ルーチンを説明するためのフローチャートである。

【０２１５】

ここでは、図５５に示すように、まず、前回のシミュレーション終了時のエンコーダ出力ｇ（ϕ_0）を出力し（ステップＳ１８０１）、今回のシミュレーション終了時の電気角φと前回の電気角ϕ_0との差分$\phi-\phi_0$が、$\phi-\phi_0>\pi/2$であるか否か（ステップＳ１８０２）、あるいは、$\phi_0-\phi>\pi/2$であるか否か（ステップＳ１８０４）が判定される。

【０２１６】

$\phi-\phi_0>\pi/2$の場合（ステップＳ１８０２のＹＥＳルート）、ステップＳ１８０３に進んで、ϕ_0にπ／２が加算され、ステップＳ１８０６において５０μｓｅｃ待機した後、ステップＳ１８０１に戻って新たなϕ_0に対するｇ（ϕ_0）が出力される。$\phi_0-\phi>\pi/2$の場合（ステップＳ１８０４のＹＥＳルート）、ステップＳ１８０５に進みϕ_0からπ／２が減算され、その後は同様にステップＳ１８０６において５０μｓｅｃ待機した後、ステップＳ１８０１に戻って新たなϕ_0に対するｇ（ϕ_0）が出力される。

【０２１７】

つまり、ここでは$|\phi-\phi_0|>\pi/2$のときは、$|\phi-\phi_0|\leqq\pi/2$になるまで、５ｋＨｚの周波数を上限として（ステップＳ１８０６）、φがπ／２ずつ変化したときの離散的な周期信号が出力される。

$|\phi-\phi_0|\leqq\pi/2$に達すると（ステップＳ１８０２，Ｓ１８０４のＮＯルート）、ステップＳ１８０７に進み、今回のシミュレーションにおけるエンコーダの最終値であるｇ（φ）が出力され、次回のシミュレーションのために今回の

シミュレーションにおけるϕが新たなϕ_0に置き換えられる（ステップＳ１８０８）。

【０２１８】

このエンコーダ信号は、組込みソフトウェア開発部４０とは独立して動作するカウンタ回路によって受信されることを想定しており、したがってエンコーダ信号発信時に組込みソフトウェア開発部４０が停止していても問題はない。

なお、図５５に示すフローチャートは、パルス信号をｇ（ϕ）の１つのみ出力するタイプのエンコーダを対象として表現したものであるが、ｇ（ϕ）の他にｇ（$\phi+\pi/2$）を出力することにより、Ａ相，Ｂ相の二相信号を出力するタイプのエンコーダも表現することができる。

【０２１９】

また、ｇ（ϕ）のほかに$\phi-\phi_0$の正負の符号をオン／オフで出力することにより、アップ／ダウン信号を出力するタイプのエンコーダも表現することができる。

ここで、上記のシミュレーションに先立って、パルスの波形〔図５４（Ａ）～図５４（Ｅ）参照〕や、分解能や、電気角ϕがゼロになる位置をあらわすオフセットを入力することにより、エンコーダの特性がユーザにより定義される。

【０２２０】

図５６は、図４９に示す処理ルーチンのステップＳ１４０６で実行されるエンコーダ処理ルーチンを説明するためのフローチャートである。

ここでは、エンコーダの故障についても考慮されており、エンコーダに故障があったときの３次元機構モデルのシミュレーションのために、予めユーザにより以下の故障フラグ０～６が設定される。

【０２２１】

故障フラグ０：正常

故障フラグ１：Ａ相がＶmaxで故障

故障フラグ２：Ａ相がＶminで故障

故障フラグ３：Ｂ相がＶmaxで故障

故障フラグ４：Ｂ相がＶminで故障

故障フラグ５：アップ／ダウン信号故障（Ｈｉｇｈに固定）

故障フラグ６：アップ／ダウン信号故障（Ｌｏｗに固定）

図５６に示すように、まず、ステップＳ１９０１で、変位θが電気角φに変換され、次いで、故障フラグが１であるか否か（ステップＳ１９０２）、２であるか否か（ステップＳ１９０３）が判定され、それらの何れでもないときは、ステップＳ１９０４に進んで、図５５に示すｇ（φ）出力処理ルーチンが実行される。

【０２２２】

一方、ステップＳ１９０２において故障フラグが１であると判定されるとステップＳ１９０５に進み、Ｖmaxで故障していることに対応してｇ（π／４）が出力される一方、ステップＳ１９０３において故障フラグが２であると判定されるとステップＳ１９０６に進み、Ｖminで故障していることに対応してｇ（３π／４）が出力される。

【０２２３】

つづいて、ステップＳ１９０７では、このエンコーダが２相出力タイプのエンコーダであるか否か、ステップＳ１９０８では、このエンコーダがアップ／ダウン信号を出力するタイプのエンコーダであるか否かが判定され、いずれでもない場合、つまり単純にｇ（φ）のみを出力するタイプのエンコーダのときはそのままこのルーチンを抜ける。

【０２２４】

ステップＳ１９０７においてこのエンコーダが２相出力タイプのエンコーダであると判定されると、ステップＳ１９０９において故障フラグが３であるか否か、ステップＳ１９１０において故障フラグが４であるか否かが判定される。それらのいずれでもなかったときは、ステップＳ１９１１に進み、図５５におけるφをφ＋π／２に置き換えた出力処理ルーチンが実行される。

【０２２５】

ステップＳ１９０９において故障フラグが３であると判定されるとステップＳ１９１２に進み、Ｂ相がＶmaxで故障していることに対応してｇ（π／４）が出力され、これと同様に、ステップＳ１９１０において故障フラグが４であると判

定されるとステップＳ１９１３に進み、Ｂ相がＶminで故障していることに対応してｇ（３π／４）が出力される。

【０２２６】

また、ステップＳ１９０８においてこのエンコーダがアップ／ダウン（Ｕ／Ｄ）信号出力タイプのエンコーダであると判定されると、ステップＳ１９１４に進み故障フラグが５であるか否かが判定され、ステップＳ１９１５では故障フラグが６であるか否かが判定される。

それらのいずれでもないときはステップＳ１９１６に進みそのエンコーダの移動方向が判定され、アップ方向（Ｕルート）のときはステップＳ１９１７に進んで‘Ｈｉｇｈ’が出力され、ダウン方向（Ｄルート）のときはステップＳ１９１８に進んで‘Ｌｏｗ’が出力される。

【０２２７】

一方、ステップＳ１９１４において故障フラグが５であると判定されるとステップＳ１９１７に進み、エンコーダの移動方向のいかんによらず‘Ｈｉｇｈ’が出力され、これと同様に、ステップＳ１９１５において故障フラグが６であると判定されるとステップＳ１９１８に進み、エンコーダの移動方向のいかんによらず‘Ｌｏｗ’が出力される。

【０２２８】

このようにしてエンコーダを表現することができるとともに、エンコーダの故障を表現することができる。ただし、エンコーダの場合、オン／オフセンサの場合と異なり、処理に入った直後に故障中を示すのではなく、センサ信号を出力する際に故障フラグの判定や処理が行なわれる。

図５７は、ポテンショメータの関数値の一例を説明するための図である。ここでは、ポテンショメータが、図５４（Ｄ）に示す三角波を出力する、分解能の極端に低いエンコーダであるとして実現している。例えば図５３に示す波長Ｌとして３６０度を設定すれば、図５７に示すように、θmax－θmin＝１８０度の変位で出力電圧Ｖを最小値Ｖminから最大値Ｖmaxまで変換するポテンショメータを表現することができる。

【０２２９】

図５８は、図４９に示すセンサ処理ルーチンのステップＳ１４０５で実行されるポテンショメータ前処理ルーチンを説明するためのフローチャートである。

ポテンショメータの場合、図５７に実線で示すように、θmin以下でのθでは出力電圧Ｖは最低値に固定され、θmax以上のθでは出力電圧Ｖは最大値に固定されることがあり、この図５８ではそのようなポテンショメータを実現している。

【０２３０】

即ち、図５８に示すように、ステップＳ２００１においてθ＞θmaxを満足するか否かが判定され、それを満足するときはステップＳ２００２に進んでθがθmaxに固定される。

また、これと同様に、ステップＳ２００３においてθ＜θminを満足するか否かが判定されθ＜θminを満足するときはステップＳ２００４に進んでθがθminに固定される。

【０２３１】

ポテンショメータの場合、このような前処理が行なわれた後、図５６に示すエンコーダ処理が実行される（図４９参照）。

このタイプのポテンショメータの場合、ユーザは、θmin，θmaxおよび，Ｖmin，Ｖmaxを入力すると、システムは、その入力値を自動的に三角波ｆ（θ）の波長，振幅，オフセットに変換する。

【０２３２】

なお、ストッパがないポテンショメータなど、出力が周期的に変化するポテンショメータの場合は、分解能が低い三角波や鋸波を出力するエンコーダとして取り扱われる。

図５９は、図４５のステップＳ１００５において実行されるΔＴ決定処理ルーチン（シミュレーション間隔決定処理ルーチン）を説明するためのフローチャートである。

【０２３３】

この図５９に示すように、ここでは、まず、ステップＳ２１０１において、性能劣化（応答遅れあるいはチャタリング）を起こしたセンサが存在するか否かが

判定され、そのようなセンサが存在しないときは、図４６に示す処理で求めた部品間の最短距離Ｄに応じたΔＴが決定される（ステップＳ２１０２）。一方、性能劣化を起こしたセンサが存在するときは、ΔＴとしてとり得る値の最小値が設定される（ステップＳ２１０３）。

【０２３４】

ここで、性能劣化を起こしたセンサが存在するときにΔＴを最小値に設定するのは、性能劣化の表現精度を高め、正確なシミュレーションを行なうためである。なお、正常なセンサの場合、遅れ時間はゼロに設定されているため、問題なく動作する。

図６０（Ａ）および図６０（Ｂ）は、干渉チェックによる関節可動範囲の検索法を説明するための図である。

【０２３５】

まず、ユーザによって、どの方向の制限値を設定したいのかということと、どの部品とどの部品とが接触することにより制限が起きるのかということが選択され、その後、関節可動範囲の検索が行なわれる。

この関節可動範囲の検索においては、関節の姿勢ｘを初期値x_0からΔｘずつ増加させ、その度に干渉チェックを行なう。すると、図６０（Ａ）に示すように、ｘが$x_0+n\Delta x$では干渉が発生せず、次の$x_0+(n+1)\Delta x$で干渉が起こる位置が判明する。

【０２３６】

続いて、例えばΔｘ’＝Δｘ／１０などのようにΔｘよりも小さい移動量Δｘ’を用いて、図６０（Ｂ）に示すように、ｘを$x_0+\Delta x$からΔｘ’ずつ増加させることにより、ｘが$x_0+n\Delta x+n'\Delta x'$では干渉が発生せず、$x_0+n\Delta x+(n'+1)\Delta x'$で干渉が発生する位置が判明する。

このような処理を数回繰り返すことにより、高速に且つ所望の精度でｘの制限値$x_0+n\Delta x+n'\Delta x'+n''\Delta x''$……を求めることができる。ただし、ユーザが指定した二つの部品が、関節をどのように動かしていても干渉しない位置関係であることも考えられるので、一回目の干渉位置検索の際には検索数ｎに上限を設け、無限ループを防ぐことが好ましい。

【０２３７】

図６１（Ａ）～図６１（Ｃ）は、干渉チェックによるカム関係の検索方法を説明するための図、図６２は、干渉チェックによるカム関係検索ルーチンを説明するためのフローチャートである。

ここでは、例えば図６１（Ａ）に示すごとく、予め、ユーザにより、駆動部品Ｐ１，受動部品Ｐ２，駆動部品Ｐ１の初期姿勢，受動部品Ｐ２の初期姿勢および駆動部品Ｐ１の最終姿勢が指定される。

【０２３８】

図６２に示すルーチンでは、まず、ϕ，ψに、それぞれ、駆動部品Ｐ１の初期姿勢および受動部品Ｐ２の初期姿勢が設定され（ステップＳ２２０１）、次いで、受動部品Ｐ２の非接触姿勢が検索されてψに格納される（ステップＳ２２０２）。

この後、ステップＳ２２０３では、今回求めた、駆動部品Ｐ１の姿勢ϕと、受動部品Ｐ２の非接触姿勢ψとのペアが登録され、例えば図６１（Ｂ）に示すごとく駆動部品Ｐ１の姿勢ϕが$\Delta\phi$だけ移動される（ステップＳ２２０４）。

【０２３９】

そして、駆動部品Ｐ１の姿勢ϕが最終姿勢になるまで（ステップＳ２２０５でＹＥＳ判定となるまで）、$\Delta\phi$ずつ移動された駆動部品Ｐ１に対する受動部品Ｐ２の非接触姿勢ψ〔図６１（Ｃ）参照〕を検索・演算する処理（ステップＳ２２０２）と、姿勢ϕ，ψの登録処理（ステップＳ２２０３）と、駆動部品Ｐ１の移動処理（ステップＳ２２０４）とが繰り返される。

【０２４０】

図６３は、図６２に示すカム関係検索ルーチンのステップＳ２２０２で実行される受動部品Ｐ２の非接触姿勢検索ルーチンを説明するためのフローチャートである。

この図６３に示すように、まず、ステップＳ２３０１において、受動部品Ｐ２の現在の姿勢ψがψ_0に設定され、カウンタによるカウント値ｎに初期値ゼロが設定される。

【０２４１】

ステップＳ２３０２では駆動部品Ｐ１と受動部品Ｐ２とが干渉しているか否かが判定され、干渉していないときは、現在の姿勢φが非接触姿勢となる（ステップＳ２３１５）。

ステップＳ２３０２で干渉していると判定されると、ステップＳ２３０３においてカウント値ｎが１だけインクリメントされ、受動部品Ｐ２の姿勢φとして$\phi_0+n\Delta\phi$が設定され（ステップＳ２３０４）、再度干渉しているか否かが判定される（ステップＳ２３０５）。まだ干渉していたときは、今度は受動部品Ｐ２の姿勢φとして$\phi_0-n\Delta\phi$が設定され（ステップＳ２３０６）、干渉しているか否かがもう一度判定される（ステップＳ２３０７）。まだ干渉していたときは、ステップＳ２３０８に進み、カウント値ｎがｎの上限か否か判定され、ｎの上限であったときは検索失敗とされる（ステップＳ２３０９）。カウント値ｎがまだ上限に達していなかったときは、ステップＳ２３０３に戻りカウント値ｎが１だけインクリメントされ、以下同様にして干渉チェックが行なわれる。

【０２４２】

ステップＳ２３０５で干渉していないと判定されると、ステップＳ２３１０に進み、φがΔφだけデクリメントされ、Δφが適当な整数Ｎで除算され、その除算結果がΔφとして設定され、Δφが前よりも小さい値に設定される（ステップＳ２３１１）。

次いで、φが、上記のようにして小さな値に設定されたΔφだけインクリメントされながら（ステップＳ２３１３）、干渉しているか否かが判定され（ステップＳ２３１２）、干渉した状態から干渉していない状態に変化すると、ステップＳ２３１４に進んで所定の精度まで検索したか否かが判定され、まだ精度が不十分なときはステップＳ２３１０に戻ってφからΔφが減算され、ステップＳ２３１１においてΔφとしさらに小さい値が設定され、φをさらに小さい値であるΔφずつインクリメントしながら干渉チェックが行なわれる（ステップＳ２３１２，Ｓ２３１３）。

【０２４３】

このようにして所定の精度まで検索が行なわれると（ステップＳ２３１４）、これにより非接触姿勢が定まる（ステップＳ２３１５）。

ステップＳ２３０７において干渉していないと判定されたときも、ステップＳ２３１０～Ｓ２３１４の処理と同様の処理が実行され、受動部品Ｐ２の非接触姿勢が求められる。なお、ステップＳ２３０７において干渉していないと判定されたときは、ステップＳ２３１０に相当するステップにおいてφはΔφだけインクリメントされ、ステップＳ２３１３に相当するステップにおいてφはΔφだけデクリメントされる。

【０２４４】

図６４は、図６２に示すカム関係検索ルーチンのステップＳ２２０２で実行される受動部品の非接触姿勢検索ルーチンを説明するためのフローチャートである。この図６４に示すルーチンは、受動部品に重力あるいはバネ付勢力が作用している場合に実行されるルーチンである。受動部品に作用する重力あるいはバネの付勢方向はユーザにより予め設定される。つまり、図６４は、負方向に力が加わっている場合のフローチャートを示している。

【０２４５】

図６４に示すルーチンでは、まず、ステップＳ２４０１において、受動部品の現在の姿勢φがϕ_0に設定され、カウンタのカウント値ｎに初期値ゼロが設定される。

ステップＳ２４０２では、駆動部品と受動部品とが現在干渉しているか否かが判定される。既に干渉しているときは、ステップＳ２４０３においてカウント値ｎを１だけインクリメントする。そして、この場合、駆動部品の今回の姿勢変化によって受動部品が干渉したものであるため、受動部品の姿勢を重力あるいはバネ付勢力に逆らって変化させ（ステップＳ２４０４～Ｓ２４０７）、干渉していない姿勢をみつけた後、φからΔφを減算することで（ステップＳ２４０８）、一旦干渉した状態に戻す。ここで、ステップＳ２４０３～Ｓ２４０８の処理は、それぞれ、図６３のステップＳ２３０３～Ｓ２３０５，Ｓ２３０８，Ｓ２３０９，Ｓ２３１０に対応している。

【０２４６】

一方、ステップＳ２４０２で干渉していないと判定されるときは、ステップＳ２４０９においてカウント値ｎを１だけインクリメントする。そして、この場合

、駆動部品の今回の姿勢変化によって受動部品が干渉していない状態に変化したものであるため、重力あるいはバネ付勢力の作用をシミュレートして受動部品が能動部品に干渉した状態を見つける（ステップＳ２４１０～Ｓ２４１３）。ここで、ステップＳ２４０９～Ｓ２４１３の処理は、それぞれ、図６３のステップＳ２３０３，Ｓ２３０６～Ｓ２３０９に対応している。

【０２４７】

そして、ステップＳ２４０８やＳ２４１１からステップＳ２４１４に進んだとき、Δφのきざみでぎりぎり干渉した状態にあり、Δφを適当な整数Ｎで除算しその除算結果をΔφとして設定することにより、Δφが前よりも小さい値に設定される。この後、φをΔφずつインクリメントしながら非干渉姿勢が検索される（ステップＳ２４１５，Ｓ２４１６）。

【０２４８】

ステップＳ２４１７では所定の精度まで検索が行なわれたか否かが判定され、未だ精度が不十分なときはステップＳ２４０８に進んでφからΔφが減算されて干渉した状態に戻され、ステップＳ２４１４においてΔφが整数Ｎで除算されてΔφとしてさらに小さい値が設定されて検索が繰り返される。

このようにして所定の精度まで検索が行なわれると（ステップＳ２４１４）、非接触位置が決定される（ステップＳ２４１８）。ここで、ステップＳ２４０８，Ｓ２４１４～Ｓ２４１８の処理は、それぞれ、図６３のステップＳ２３１０～Ｓ２３１５に対応している。

【０２４９】

これにより、重力あるいはバネ付勢力が作用しない場合であっても作用する場合であっても、カム関係を決定することができる。

図６５（Ａ）～図６５（Ｃ）は、駆動部材Ｐ１側の溝Ｐ１０とその溝Ｐ１０に挿入されて駆動される受動部材Ｐ２側のピンＰ２０との溝関係を設定する手法を説明するための図であり、図６５（Ｂ）および図６５（Ｃ）は、図６５（Ａ）の要部を拡大して示したものである。

【０２５０】

図６５（Ａ）および図６５（Ｂ）に示すような溝関係を設定しようとしたとき

、ピンＰ２０は溝Ｐ１０に常に接触した状態にあるため、非干渉位置を求めることはできず、このような溝関係を、前述したカム関係の検索法を用いて求めることは困難である。

そこで、ここでは、図６５（Ｃ）に示すように、ピンＰ２０の中心に半径ｒの微小円筒Ｐ２１を一時的に取り付け、この微小円筒Ｐ２１と溝Ｐ１０との間の距離ｄを測定することで溝関係を求める。即ち、図６３のステップＳ２３０２，Ｓ２３０５，Ｓ２３０７，Ｓ２３１２において、「干渉しているか?」の判定に代えて「ｄ＜Ｒ－ｒか?」の判定を行なうことにより、溝関係を検索することができる。

【０２５１】

さらに、以下に、本実施形態の３Ｄモデルシミュレータ３０における、各種機構（ローラ，ラチェット，モータ，センサ等）のシミュレーション手法について、図６６～図８２を参照しながら説明する。

図６６は、ギアやローラをリンクとして有する場合のリンク姿勢移動ルーチンを説明するためのフローチャートであり、この図６６に示すルーチンは、例えば、前述したモータ処理ルーチン（図４６のステップＳ１１０７）で実行されるリンク姿勢移動サブルーチン（図４７参照）に対応するものである。また、図６７は関節値移動サブルーチンを説明するためのフローチャート、図６８は関節値初期化サブルーチンを説明するためのフローチャートである。なお、以下では、リンクのことを関節と呼ぶ場合がある。

【０２５２】

ここで、ローラはギアの中の一機能であり、以下に説明するルーチンでは、図６６～図６８に示すフローチャートのごとく、図６７に示す関節値移動サブルーチンを再帰的に呼び出して実行することにより、現関節の動作に従って動作する従動関節（受動関節）を移動させていく。その移動途中において、従動関節の一つが制限値に到達した場合には、その関節値を予め設定した初期値に戻すために、図６８に示す関節値初期化サブルーチンを再帰的に呼び出して実行している。

【０２５３】

図６６に示すように、リンク姿勢移動（例えばモータリンクの姿勢移動）を受

けて本ルーチンが呼び出されると、モータリンクの移動に伴って現関節を移動させるべく、図６７に示す関節値移動サブルーチンを呼び出して実行する（ステップＳ２５０１）。そして、ステップＳ２５０１での関節値移動サブルーチンからの戻り値が“ＴＲＵＥ”か否かを判定し（ステップＳ２５０２）、戻り値が“ＴＲＵＥ”である場合（ＹＥＳルート）には、現関節について、図６８に示す関節値初期化サブルーチンを呼び出して実行してから（ステップＳ２５０３）、本ルーチンの処理を終了する。ステップＳ２５０２で戻り値が“ＴＲＵＥ”でないと判定された場合（ＮＯルート）には、そのまま、本ルーチンの処理を終了する。

【０２５４】

関節値移動サブルーチンでは、図６７に示すように、関節値の移動を行なってから（ステップＳ２５１１）、その関節値が制限値に到達したか否かを判定する（ステップＳ２５１２）。制限値に到達した場合（ＹＥＳルート）、戻り値として“ＴＲＵＥ”が出力される（ステップＳ２５１３）。制限値に到達していない場合（ＮＯルート）、未処理の従動関節があるか否かを判定する（ステップＳ２５１４）。未処理の従動関節がない場合（ＮＯルート）、戻り値として“ＦＡＬＳＥ”が出力される（ステップＳ２５１５）。

【０２５５】

また、ステップＳ２５１４で未処理の従動関節があると判定された場合（ＹＥＳルート）、ギア，カム関係から従動関節の移動量を計算してから（ステップＳ２５１６）、その従動関節について関節値移動サブルーチンを呼び出して実行し（ステップＳ２５１７）、このステップＳ２５１７で実行されたサブルーチンからの戻り値が“ＴＲＵＥ”か否かを判定し（ステップＳ２５１８）、戻り値が“ＴＲＵＥ”である場合（ＹＥＳルート）には、戻り値として“ＴＲＵＥ”が出力される（ステップＳ２５１９）。戻り値が“ＴＲＵＥ”ではないと判定された場合（ＮＯルート）、ステップＳ２５１４に戻る。

【０２５６】

関節値初期化サブルーチンでは、図６８に示すように、関節値として旧関節値を設定してから（ステップＳ２５２０）、未処理の従動関節があるか否かを判定し（ステップＳ２５２１）、未処理の従動関節がない場合（ＮＯルート）、この

関節値初期化サブルーチンを呼び出したステップの処理に戻る一方、未処理の従動関節がある場合（ＹＥＳルート）、その従動関節について、関節値初期化サブルーチンを呼び出して実行してから（ステップＳ２５２２）、ステップＳ２５２１に戻る。

【０２５７】

図６９（Ａ）～図６９（Ｃ）はギアやローラの動作について説明するための図である。

ここでは、図６９（Ａ）に示すように、駆動部品としてのドラムを回転させることにより、従動部品（受動部品）としての２つのスライダ１，２に駆動力が伝達され、これらのスライダ１，２が水平移動する機構について説明する。

【０２５８】

図６９（Ａ）に示す機構がギアで構成されている場合、図６９（Ｂ）に示すように、ドラムの反時計回りの回転に伴いスライダ１の左端が左壁面に当たると（即ち、スライダ１が制限値に到達すると）、それ以上、ドラムやスライダ２は動かなくなる。

一方、図６９（Ａ）に示す機構において、ドラムとスライダ１との間のギアが滑りを許可するローラで構成されている場合、スライダ１が制限値に到達して動かなくなっても、図６９（Ｃ）に示すように、ドラムは反時計回りに回動し続け、スライダ２は依然として左方向へ水平移動することになる。

【０２５９】

図７０は、ローラをリンクとして有する場合の関節移動サブルーチンの変更部分を説明するためのフローチャートである。

このようなローラをシミュレータ３０でシミュレートする場合、図６７のステップＳ２５１８とステップＳ２５１９との間に、図７０に示すように、従動関節がローラか否かを判定するステップＳ２５３０が追加され、ローラである場合（ＹＥＳルート）、ステップＳ２５１４に戻る一方、ローラでない場合（ＮＯルート）、ステップＳ２５１９へ進むようにする。つまり、処理対象の従動関節がローラかどうかを調べ、ローラである場合には、制限値処理を行なった後、その従動関節よりも上流側に制限値情報を流さないようにする。

【０２６０】

次に、ラチェットをシミュレータ３０でシミュレートする手法について簡単に説明する。

ラチェットとは、自転車のフリーやラチェットレンチのように、ユーザが指定した方向には動力を伝達するが、その方向とは逆方向には動力伝達しないものである。

【０２６１】

このようなラチェットを、シミュレータ３０で実現（シミュレート）する手法は、図４７で説明したクラッチをシミュレートする手法とほぼ同様である。シミュレートすべき機構にラチェットが存在する場合、ラチェットを成す駆動部品の動く方向がユーザの指定した方向であれば動力を伝達させて、受動部品についての移動処理（関節値移動処理）を実行する一方、逆方向であれば動力を伝達させないようにする。

【０２６２】

図７１（Ａ）～図７１（Ｆ）は、機構モデルとしての昇降機とその昇降機を成すリンクのツリー構造とを説明するための図、図７２は、各リンクに持たせる、親子関係の切替情報を示す図、図７３は、親切替チェックサブルーチンを説明するためのフローチャートである。

機構モデルを構成する各リンク（関節）は、一般に、ツリー状になっており、任意のリンクを動かすと、そのリンクの子（チャイルド）は全て同様に動くことになる。このようなデータ構造では、図７１（Ａ）～図７１（Ｆ）に示すような昇降機を表現することが難しい。そこで、本実施形態では、クラッチと同様、何らかの条件（例えば第３の関節値）によって親を切り替える機能を追加する。

【０２６３】

図７１（Ａ）～図７１（Ｃ）に示す昇降機は、水平に移動可能な車Ｃを、上下方向に移動可能なテーブルＴにより、異なる高さの土台Ｂ，Ｂの間で移動させるものである。図７１（Ａ）に示す状態の昇降機は、図７１（Ｄ）に示すように、車Ｃおよびテーブルを土台Ｂの子とするツリー構造で表される。また、図７１（Ｂ）に示す状態の昇降機は、図７１（Ｅ）に示すように、車Ｃをテーブルの

土台Ｂの子としテーブルＴを土台Ｂの子とするツリー構造で表される。そして、図７１（Ｃ）に示す状態の昇降機は、図７１（Ｆ）に示すように、車Ｃおよびテーブルを土台Ｂの子とするツリー構造で表される。

【０２６４】

このように親子関係が切り替わる機構（昇降機等）についてシミュレーションを行なう場合には、各リンク（関節）に図７２に示すような親子関係の切替情報を予め持たせておき、シミュレータ３０によるシミュレーションに際し、関節値が変化した場合（関節値移動後）、図７２に示すような情報に基づいて、その変化に応じて親子関係に影響の出る可能性のある子部品について、親を再計算し、親の切替を行なう。

【０２６５】

例えば、図６７のステップＳ２５１１で関節値移動を行なった後、親子切替部品の有無を判定し、親子切替部品がある場合には、図７３に示す親切替チェックサブルーチンを呼び出して実行する。

この図７３に示すように、親切替チェックサブルーチンでは、まず、未処理の親部品候補が存在するか否かを判定し（ステップＳ２５３１）、存在しない場合（ＮＯルート）、このサブルーチンの処理を終了する一方、存在する場合（ＹＥＳルート）、未処理の、親が切り替わる条件が存在するか否かを判定する（ステップＳ２５３２）。

【０２６６】

未処理の条件が存在する場合（ステップＳ２５３２のＹＥＳルート）、その条件が成り立つか否か判定し（ステップＳ２５３３）、成立する場合（ＹＥＳルート）、ステップＳ２５３２に戻る一方、成立しない場合（ＮＯルート）、ステップＳ２５３１に戻る。そして、ステップＳ２５３２で未処理の条件が存在しないと判定された場合（ＮＯルート）、親の切替を行なってから（ステップＳ２５３４）、このサブルーチンの処理を終了する。

【０２６７】

次に、相対クラッチをシミュレータ３０でシミュレートする手法について簡単に説明する。

クラッチが、何らかの条件（例えば第３の関節の関節値）によってギア，カム関係の動力伝達のオン／オフを決めるものであるに対し、相対クラッチとは、動力伝達のオン／オフを、任意の２つの部品の相対位置によって決めるもので、親子関係切替を用いた並べ替え機構などを実現する。

【０２６８】

一般のクラッチのシミュレーションに際して「クラッチ部品の姿勢はｏｎの位置か」を調べていたのに対し（図４７のステップＳ１２０４参照）、上述のような相対クラッチのシミュレーションに際しては、「部品Ａから見て部品Ｂがｏｎの位置か」を調べる。それ以外の処理は、一般のクラッチのシミュレーションと全く同様である。

【０２６９】

ここで、部品Ａから見た部品Ｂの位置は、R_A^{-1}（$P_A - P_B$）により求めることができる。ただし、R_Aは部品Ａのローカル座標の絶対座標からの回転行列、P_A，P_Bはそれぞれ絶対座標から見た部品Ａ，Ｂの位置である。

次に、ＤＣモータをシミュレータ３０でシミュレートする手法について、図７４および図７５を参照しながら説明する。なお、図７４は、ＤＣモータのシミュレーション手順を説明するためのフローチャート、図７５は、ＤＣモータの入力値とモータ指令値との対応関係の一例を示す図である。

【０２７０】

ＤＣモータのシミュレーションを行なう際には、ＤＣモータの入力値とモータ指令値との対応関係を、例えば図７５に示すようなテーブルとして予め登録しておく。このテーブルは、図２６において前述した対応表と同様のものである。ただし、図７５に示すテーブルでは、ＤＣモータの２つのポート（ポート番号１１，１２）に入力される値（アクチュエータ指令信号）と、目標速度(deg/sec)および整定時間(msec)との対応関係が設定されている。

【０２７１】

図７５のテーブルを用いた場合、ポート番号１１，１２にそれぞれ０，１が入力されると、目標速度＋１００deg/secおよび整定時間５００secがモータ指令値として出力され、ポート番号１１，１２にそれぞれ１，０が入力されると、目標

速度－１００deg/secおよび整定時間５００secがモータ指令値として出力される。

【０２７２】

このようなテーブルを設定・登録しておいてから、図７４に示す手順でシミュレータ３０によりＤＣモータのシミュレーションを行なう。即ち、組込みソフトウェア開発部４０からのアクチュエータ指令信号を、ポート番号の状態入力として認識してから（ステップＳ２５４１）、その入力値を図７５のテーブルによりモータ指令値（目標速度および整定時間）に変換する（ステップＳ２５４２）。

【０２７３】

そして、既に算出されている今回のシミュレーション時間T_S（＝Δｔ）を入力し（ステップＳ２５４３）、今回のシミュレーション時間T_S（＝Δｔ）でのＤＣモータの回転角を計算する（ステップＳ２５４４）。このとき、回転角は、目標速度×シミュレーション時間として算出できるが、モータ指令値が入力されてからのトータルのシミュレーション時間が整定時間以下であれば、前述したごとく、式（４）を用いて回転角が算出される。この後、そのＤＣモータに連結した全ての関節（リンク）の関節値を計算して処理を終了する（ステップＳ２５４５）。

【０２７４】

次に、ステッピングモータをシミュレータ３０でシミュレートする手法について、図７６～図７８を参照しながら説明する。なお、図７６は、ステッピングモータのシミュレーション手順を説明するためのフローチャート、図７７は、ステッピングモータの加速パターンを説明するための図、図７８は、パルス数計算サブルーチンを説明するためのフローチャートである。

【０２７５】

ステッピングモータのシミュレーションを行なう際には、例えば図７７に示すような、ステッピングモータの加速パターンＴｈ〔Ｎ〕を事前に登録しておく。この加速パターンＴｈ〔Ｎ〕としては、加速を開始してｉ番目のパルスの時間間隔がｔｉとして登録されている。

このような加速パターンＴｈ〔Ｎ〕を登録しておいてから、図７６に示す手順

でシミュレータ３０によりステッピングモータのシミュレーションを行なう。即ち、組込みソフトウェア開発部４０から、アクチュエータ指令信号であるモータ指令値（目標速度）を入力するとともに（ステップＳ２５５１）、既に算出されている今回のシミュレーション時間T_S（＝Δｔ）を入力し（ステップＳ２５５２）、今回のシミュレーション時間T_S（＝Δｔ）内に、ステッピングモータに対して出力されるパルス数を、図７８に示すパルス数計算サブルーチンにより求める（ステップＳ２５５３）。

【０２７６】

そして、ステップＳ２５５３で計算されたパルス数に基づいて、ステッピングモータの回転角を計算し（ステップＳ２５５４）、そのステッピングモータに連結した全ての関節（リンク）の関節値を計算して処理を終了する（ステップＳ２５５５）。

なお、パルス数計算サブルーチンでは、図７８に示すように、まず、パルス数ｐとして累積パルス数を、シミュレーション時間ｔとして０を初期設定しておいてから（ステップＳ２５６１）、ｔとしてｔ＋Ｔｈ〔ｐ〕を設定し（ステップＳ２５６２）、置き換え後の時間ｔがシミュレーション時間T_Sよりも小さいか否かを判定する（ステップＳ２５６３）。ｔ＜T_Sであれば（ＹＥＳルート）、ｐとしてｐ＋１を設定し（ステップＳ２５６４）、ステップＳ２５６２に戻る一方、シミュレーション時間ｔがT_Sに達すると、このパルス数計算サブルーチンを終了し、パルス数ｐを計算結果として出力する。

【０２７７】

次に、ＡＣモータをシミュレータ３０でシミュレートする手法について、図７９を参照しながら説明する。なお、図７９は、ＡＣモータのシミュレーション手順を説明するためのフローチャートである。

この図７９に示すように、シミュレータ３０によりステッピングモータのシミュレーションを行なう際には、まず、組込みソフトウェア開発部４０から、アクチュエータ指令信号であるモータ指令値（目標速度）を入力し（ステップＳ２５７１）、その目標速度に対応した整定時間（その対応関係は予め登録されている）を入力するとともに（ステップＳ２５７２）、既に算出されている今回のシミ

ュレーション時間T_S（＝Δｔ）を入力し（ステップＳ２５７３）、今回のシミュレーション時間T_S（＝Δｔ）でのＡＣモータの回転角を計算する（ステップＳ２５７４）。

【０２７８】

このとき、回転角は、目標速度×シミュレーション時間として算出できるが、モータ指令値が入力されてからのトータルのシミュレーション時間が整定時間以下であれば、回転角は、加減速を考慮して計算される。この後、そのＡＣモータに連結した全ての関節（リンク）の関節値を計算して処理を終了する（ステップＳ２５７５）。

【０２７９】

次に、マイクロスイッチをシミュレータ３０でシミュレートする手法について、図８０を参照しながら説明する。なお、図８０は、マイクロスイッチのシミュレーション手順を説明するためのフローチャートである。

この図８０に示すように、シミュレータ３０によりマイクロスイッチのシミュレーションを行なう際には、まず、マイクロスイッチをセンサとして設定した関節の関節値を入力するとともに（ステップＳ２５８１）、その関節値とセンサ値との対応関係（テーブル）を入力してから（ステップＳ２５８２）、関節値とテーブルとに基づいてセンサ値（オン／オフ）を判定し（ステップＳ２５８３）、そのセンサ値を組込みソフトウェア開発部４０に出力する（ステップＳ２５８４）。

【０２８０】

次に、エンコーダをシミュレータ３０でシミュレートする手法について、図８１を参照しながら説明する。なお、図８１は、エンコーダのシミュレーション手順を説明するためのフローチャートである。

この図８１に示すように、シミュレータ３０によりエンコーダのシミュレーションを行なう際には、まず、エンコーダをセンサとして設定した関節の関節値を入力するとともに（ステップＳ２５９１）、エンコーダの分解能やエンコーダ原点での関節値を入力してから（ステップＳ２５９２）、関節値に対応した波形を計算することにより関節値に対応したセンサ値を算出し（ステップＳ２５９３）

、算出されたセンサ値を組込みソフトウェア開発部４０に出力する（ステップＳ２５９４）。

【０２８１】

次に、ポテンショメータをシミュレータ３０でシミュレートする手法について、図８２を参照しながら説明する。なお、図８２は、ポテンショメータのシミュレーション手順を説明するためのフローチャートである。

この図８２に示すように、シミュレータ３０によりポテンショメータのシミュレーションを行なう際には、まず、ポテンショメータをセンサとして設定した関節の関節値を入力するとともに（ステップＳ２６０１）、その関節値とセンサ値との対応関係（テーブル）を入力してから（ステップＳ２６０２）、関節値とテーブルとに基づいてセンサ値を計算し（ステップＳ２６０３）、そのセンサ値を組込みソフトウェア開発部４０に出力する（ステップＳ２６０４）。

【０２８２】

このように、本発明の一実施形態としての支援システム１０によれば、３次元リアルタイムシミュレーション装置３０を中核に据えて、メカ試作品を作らなくても組込みソフトウェア（制御プログラム）の開発をメカ設計とは単独に進めることができる。つまり、メカ設計部２０による機構の設計と、その機構を制御するための組込みソフトウェアの開発（組込みソフトウェア開発部４０による開発）とをコンカレントに遂行することができ、その組込みソフトウェアの開発が効率化され、開発期間の短縮，工数の削減を実現することができる。

【０２８３】

このとき、状態遷移図・表によるタスク制御記述法と３次元リアルタイムシミュレーション装置３０とを組み合わせることにより、組込みソフトウェアを試作品なしに効率よく開発することができる。

また、メカ設計部２０と３次元リアルタイムシミュレーション装置３０とを連携させることにより、製品を試作することなくシミュレーションによってその３次元機構モデルの可動部を動かし、部品どうしが不用意に接触せず所期の動作を行なうことができるかどうか等、その機構の確認を行ないながら、メカ設計を行なうことができる。その結果として、設計の不具合を早期に発見し、部品の配置

変更や可動部を考慮した修正等を行なうことが可能になり、製品開発に要する時間の短縮化やコスト削減を実現することができる。

【０２８４】

さらに、マイコンチップ４２と３次元リアルタイムシミュレーション装置３０とを連携させることにより、機構を物理的に製作することなく、機構制御用のプログラムのデバックを行なうことができ、その組込みソフトウェアの開発の短縮化やコスト低減化に大きく寄与することになる。

なお、本発明は上述した実施形態に限定されるものではなく、本発明の趣旨を逸脱しない範囲で種々変形して実施することができる。

【０２８５】

【発明の効果】

以上詳述したように、本発明の支援システム（請求項１～１０）および支援プログラムを記録したコンピュータ読み取り可能な記録媒体（請求項１１～１８）によれば、３次元機構モデルシミュレーション部を中核に据えて、メカ試作品を作らなくても組込みソフトウェアの開発をメカ設計とは単独に進めることができる。つまり、機構の設計と、その機構を制御するための組込みソフトウェアの開発とをコンカレントに遂行することができ、その組込みソフトウェアの開発が効率化され、開発期間の短縮，工数の削減を実現することができる。

【図面の簡単な説明】

【図１】

本発明の一実施形態としての支援システムの機能的構成を示すブロック図である。

【図２】

本実施形態のコンピュータ読み取り可能な記録媒体に記録された支援プログラムの構成を示す図である。

【図３】

本発明の一実施形態としての支援システムを構築しうるコンピュータシステムを示す外観図である。

【図４】

本発明の一実施形態としての支援システムを構築しうるコンピュータシステムのハードウェア構成を示すブロック図である。

【図５】

本実施形態の支援システムの要部を示すブロック図である。

【図６】

（Ａ）および（Ｂ）は、それぞれ、メカ設計部から３次元リアルタイムシミュレーション装置へ受け渡される図形データファイルおよびアセンブリデータファイルの具体例を示す図である。

【図７】

本実施形態の支援システムの要部（３次元リアルタイムシミュレーション装置，状態遷移図・表作成編集装置および第２インタフェース部）を示すブロック図である。

【図８】

状態遷移図の表示例を示す図である。

【図９】

状態遷移表の表示例を示す図である。

【図１０】

（Ａ）～（Ｃ）はマルチタスクについて説明するためのタイムチャートである。

【図１１】

（Ａ）および（Ｂ）は優先度によるタスクのスケジューリングについて説明するための図である。

【図１２】

（Ａ）～（Ｄ）は同期タスクを用いた同期化処理について説明するためのタイムチャートである。

【図１３】

本実施形態における同期タスクを用いた同期化処理について説明するためのブロック図である。

【図１４】

本実施形態における同期化処理について説明するためのフローチャートである。

【図１５】

本実施形態のロジックアナライザにおける表示例を示す図である。

【図１６】

ロジックアナライザによる処理を含む、本実施形態でのシミュレーション処理全体の流れを説明するためのフローチャートである。

【図１７】

本実施形態の支援システムにおける３次元リアルタイムシミュレーション装置の動作概念を説明するためのの概念図である。

【図１８】

センサやモータの定義時の処理を説明するための、３次元リアルタイムシミュレーション装置の模式図である。

【図１９】

本実施形態の３次元リアルタイムシミュレーション装置によるシミュレーション処理を説明するための模式図である。

【図２０】

本実施形態の３次元リアルタイムシミュレーション装置と組込みソフトウェア開発部（同期タスク）との動作タイミングを説明するための図である。

【図２１】

本実施形態の３次元リアルタイムシミュレーション装置によるシミュレーション手順を説明するためのフローチャートである。

【図２２】

図２１に示すシミュレータの処理フローの一部分（図２１に一点鎖線で囲った部分）に代えて採用することのできる部分フローを示すフローチャートである。

【図２３】

モータとして定義されるリンク（モータリンク）の選び方を説明すべくモータリンクの一例を模式的に示す斜視図である。

【図２４】

（Ａ），（Ｂ）はセンサとして定義されるリンク（センサリンク）の選び方を説明すべくセンサリンクの一例を模式的に示す斜視図である。

【図２５】

（Ａ），（Ｂ）はセンサとして定義されるリンク（センサリンク）の選び方を説明すべくセンサリンクの他例を模式的に示す斜視図である。

【図２６】

モータの種類を説明するための図である。

【図２７】

モータを一次遅れ系とみなしたときの、モータの回転速度の初期変化を示す図である。

【図２８】

本実施形態の３次元リアルタイムシミュレーション装置における、一次遅れを考慮したモータ駆動の処理を説明するためのフローチャートである。

【図２９】

干渉チェックを用いたセンサの定義方法を説明するための図である。

【図３０】

干渉チェックを用いたセンサの定義方法を説明するための図である。

【図３１】

（Ａ）～（Ｃ）は干渉チェックを用いたセンサの定義方法を説明するための図である。

【図３２】

本実施形態において３次元リアルタイムシミュレーション装置と組込みソフトウェア開発部との間で同期をとるために送受信される信号を説明するためのタイミングチャートである。

【図３３】

本実施形態において、３次元機構モデルの動作とソフトウェア開発部の動作との同期をとるための同期化手法を説明するためのフローチャートである。

【図３４】

本実施形態において、３次元機構モデルの動作とソフトウェア開発部の動作と

の同期を実現するための、３次元リアルタイムシミュレーション装置側の処理を説明するためのフローチャートである。

【図３５】

（Ａ），（Ｂ）は本実施形態での干渉可能性の評価方法の説明図である。

【図３６】

（Ａ）～（Ｃ）は部品間距離とシミュレーション間隔との関係の各種の例を示す図である。

【図３７】

（Ａ），（Ｂ）は部品のグループ分けを説明するための図である。

【図３８】

（Ａ）～（Ｄ）は１つの３次元機構モデル内に複数のモータリンクが定義されている場合のグループ分けを説明するための図である。

【図３９】

グループ分けした部品の再グループ化を説明するための図である。

【図４０】

複数のリンクを、モータリンクの動きによって姿勢に影響を受けるリンクのグループとそのモータリンクの動きによっては姿勢に影響を受けないグループとに分けるルーチンを説明するためのフローチャートである。

【図４１】

図４０における検索ルーチンを説明するためのフローチャートである。

【図４２】

図４０および図４１に示すルーチンの動作説明用のリンク機構モデルを模式的に示す斜視図である。

【図４３】

図４２に示すリンク機構モデルのデータ構造を説明するための図である。

【図４４】

（Ａ）～（Ｄ）は、モータリンクの動きによって姿勢に影響を受けるリンクグループ内で、シミュレーション時に距離を求めるリンクの組を抽出する手法を具体的に説明するための図である。

【図４５】

本実施形態における別の実施態様の３次元リアルタイムシミュレーション装置によるシミュレーション処理全体の処理の流れを説明するためのフローチャートである。

【図４６】

モータ処理ルーチンを説明するためのフローチャートである。

【図４７】

リンク姿勢移動サブルーチンを説明するためのフローチャートである。

【図４８】

関節不良動作処理ルーチンを説明するためのフローチャートである。

【図４９】

センサ処理ルーチンを説明するためのフローチャートである。

【図５０】

オン／オフセンサ処理ルーチンを説明するためのフローチャートである。

【図５１】

センサ性能劣化処理ルーチンを説明するためのフローチャートである。

【図５２】

チャタリングによる出力予定値変更処理ルーチンを説明するためのフローチャートである。

【図５３】

エンコーダの変位（角度等）に対する出力波形の一例を示す図である。

【図５４】

（Ａ）～（Ｅ）は、エンコーダの、各種の出力波形を示す図である。

【図５５】

エンコーダの出力処理ルーチンを説明するためのフローチャートである。

【図５６】

エンコーダ処理ルーチンを説明するためのフローチャートである。

【図５７】

ポテンショメータの関数値の一例を説明するための図である。

【図５８】

ポテンショメータ前処理ルーチンを説明するためのフローチャートである。

【図５９】

シミュレーション間隔決定処理ルーチンを説明するためのフローチャートである。

【図６０】

（Ａ），（Ｂ）は干渉チェックによる関節可動範囲の検索法を説明するための図である。

【図６１】

（Ａ）～（Ｃ）は干渉チェックによるカム関係の検索方法を説明するための図である。

【図６２】

干渉チェックによるカム関係検索ルーチンを説明するためのフローチャートである。

【図６３】

受動部品の非接触姿勢検索ルーチンを説明するためのフローチャートである。

【図６４】

受動部品の非接触姿勢検索ルーチンを説明するためのフローチャートである。

【図６５】

（Ａ）～（Ｃ）は駆動部材側の溝とその溝に挿入されて駆動される受動部材側のピンとの溝関係を設定する手法を説明するための図である。

【図６６】

ギアやローラをリンクとして有する場合のリンク姿勢移動ルーチンを説明するためのフローチャートである。

【図６７】

関節値移動サブルーチンを説明するためのフローチャートである。

【図６８】

関節値初期化サブルーチンを説明するためのフローチャートである。

【図６９】

（Ａ）～（Ｃ）はギアやローラの動作について説明するための図である。

【図７０】

ローラをリンクとして有する場合の関節移動サブルーチンの変更部分を説明するためのフローチャートである。

【図７１】

（Ａ）～（Ｆ）は、機構モデルとしての昇降機とその昇降機を成すリンクのツリー構造とを説明するための図である。

【図７２】

各リンクに持たせる、親子関係の切替情報を示す図である。

【図７３】

親切替チェックサブルーチンを説明するためのフローチャートである。

【図７４】

ＤＣモータのシミュレーション手順を説明するためのフローチャートである。

【図７５】

ＤＣモータの入力値とモータ指令値との対応関係の一例を示す図である。

【図７６】

ステッピングモータのシミュレーション手順を説明するためのフローチャートである。

【図７７】

ステッピングモータの加速パターンを説明するための図である。

【図７８】

パルス数計算サブルーチンを説明するためのフローチャートである。

【図７９】

ＡＣモータのシミュレーション手順を説明するためのフローチャートである。

【図８０】

マイクロスイッチのシミュレーション手順を説明するためのフローチャートである。

【図８１】

エンコーダのシミュレーション手順を説明するためのフローチャートである。

【図８２】

ポテンショメータのシミュレーション手順を説明するためのフローチャートである。

【符号の説明】

１０　支援システム

２０　メカ設計部（機構設計部，意匠設計・３次元ＣＡＤシステム）

３０　３次元リアルタイムシミュレーション装置（３次元機構モデルシミュレーション部，３Ｄモデルシミュレータ）

３１　シミュレーション部

３２　インタフェース部

４０　組込みソフトウェア開発部

４１　状態遷移図・表作成編集装置（状態遷移図・表作成部）

４１１　タスク制御部

４１２　ユーザインタフェース部

４１３　状態遷移表示部

４１４　インタフェース部

４１５　内部メモリ

４２　マイコンチップ

４３　統合開発環境装置

５０　第１インタフェース部

５１，５２　第２インタフェース部

５３　共有メモリ

１００　コンピュータシステム

１０１　本体部

１０１ａ　フロッピィディスク装填口

１０１ｂ　ＣＤ－ＲＯＭ装填口

１０２　ＣＲＴディスプレイ

１０２ａ　表示画面

１０３　キーボード

１０４　マウス

２１０　ＣＤ－ＲＯＭ

２１１　ハードディスク

２１２　フロッピィディスク

２２０　バスライン

２２１　中央演算処理装置（ＣＰＵ）

２２２　ＲＡＭ

２２３　ハードディスクコントローラ

２２４　フロッピィディスクドライバ

２２５　ＣＤ－ＲＯＭドライバ

２２６　マウスコントローラ

２２７　キーボードコントローラ

２２８　ディスプレイコントローラ

２２９　モデム

２３０　ＲＯＭ

３００　記録媒体

３１０　支援プログラム

３１１　機構設計プログラム

３１２　３次元機構モデルシミュレーションプログラム

３１３　組込みソフトウェア開発プログラム

３１４　第１インタフェースプログラム

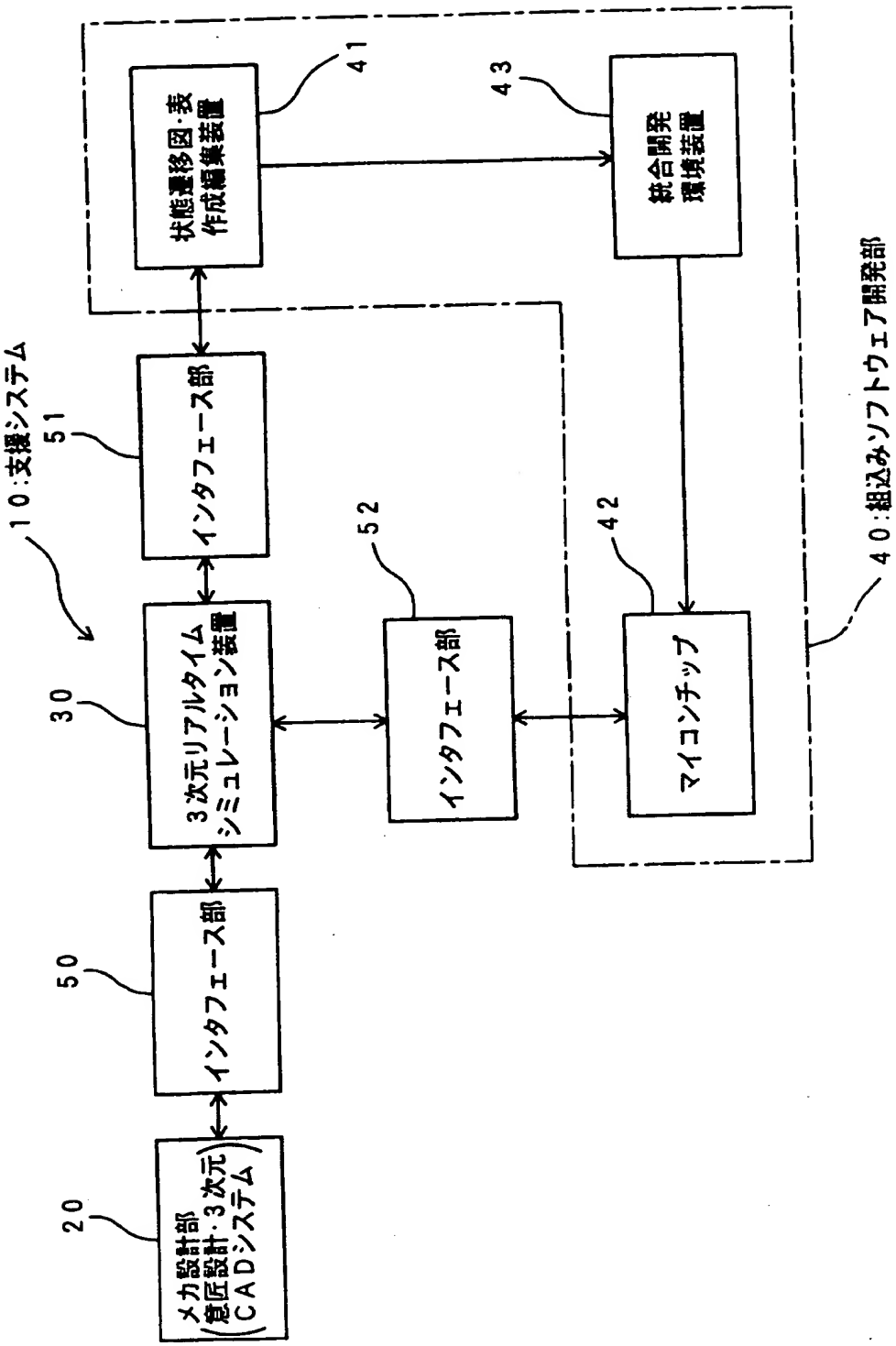
３１５　第２インタフェースプログラム

３１６　統合開発環境プログラム

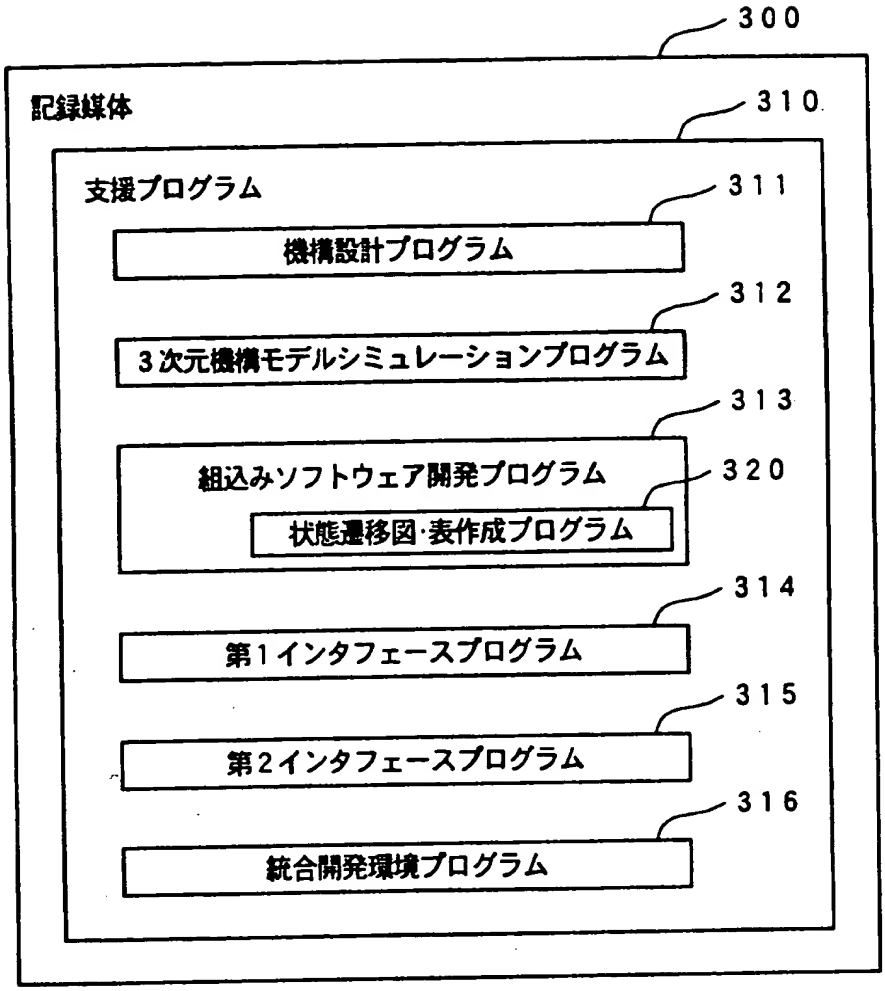
３２０　状態遷移図・表作成プログラム

【書類名】　　図面

【図１】

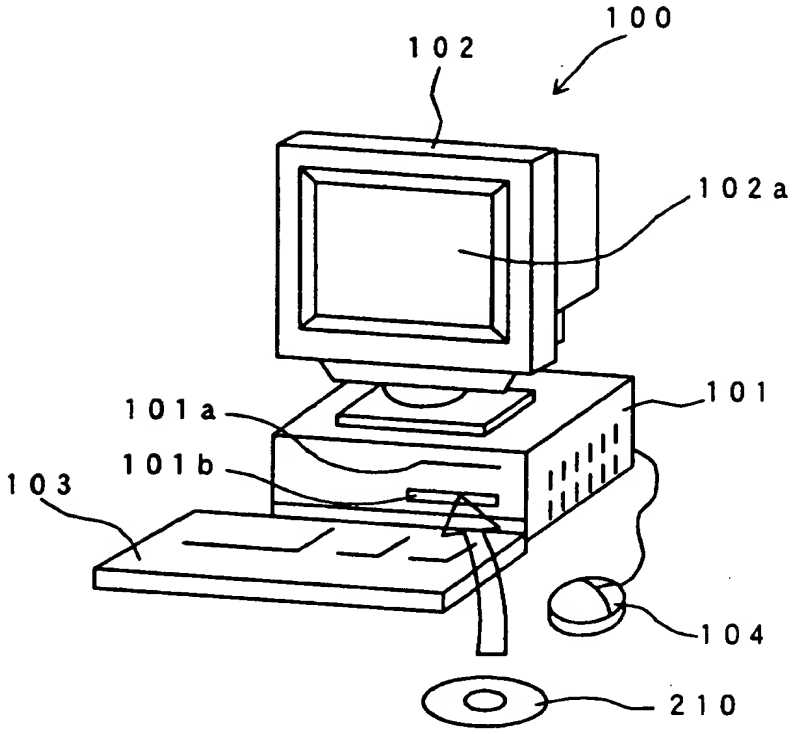

状態遷移図・表作成編集装置
41
統合開発環境装置
43
40:組込みソフトウェア開発部
10:支援システム
インタフェース部
51
3次元リアルタイムシミュレーション装置
30
インタフェース部
52
マイコンチップ
42
インタフェース部
50
メカ設計部(意匠設計・3次元CADシステム)
20

【図２】

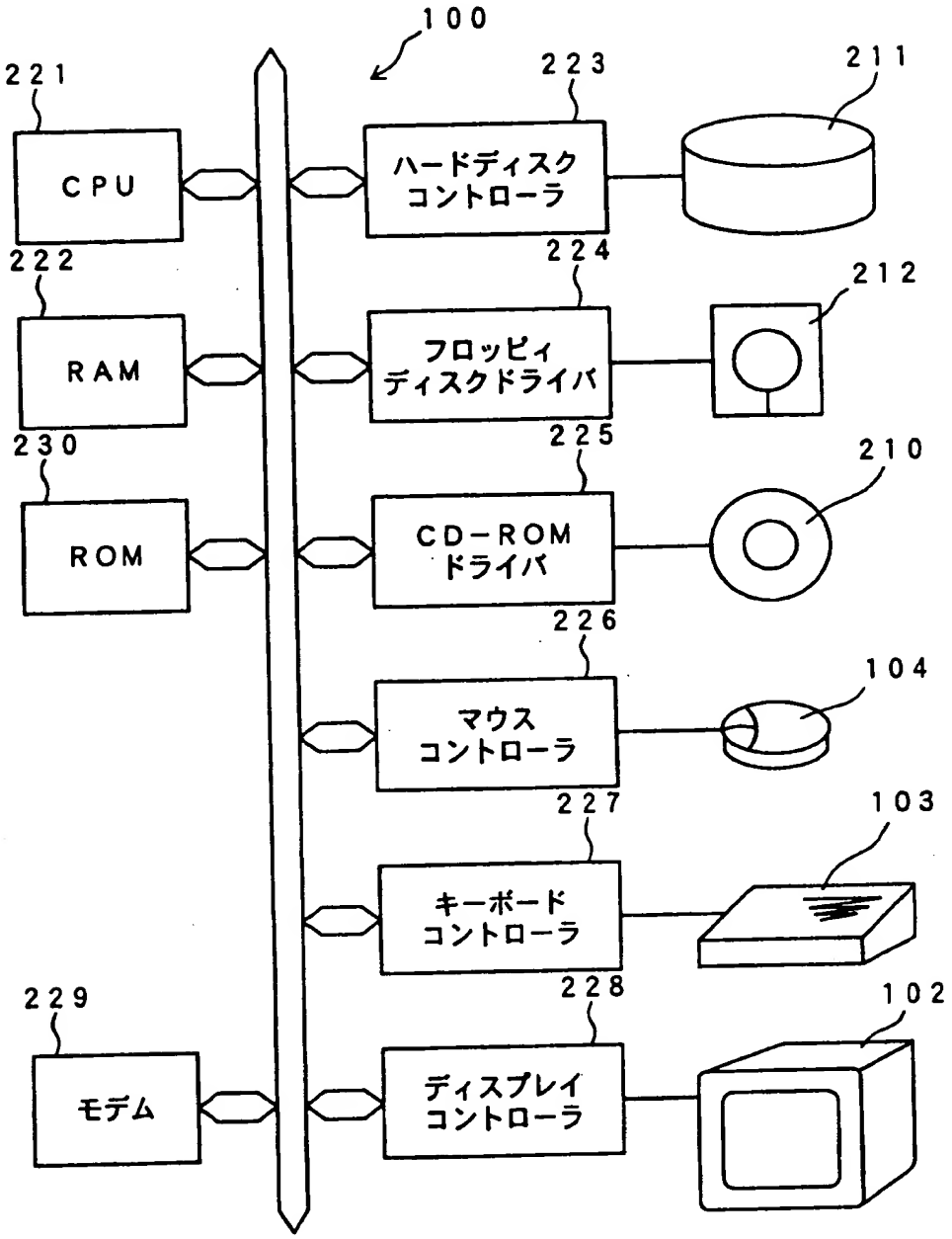


出証特２０００－３０８０２４８

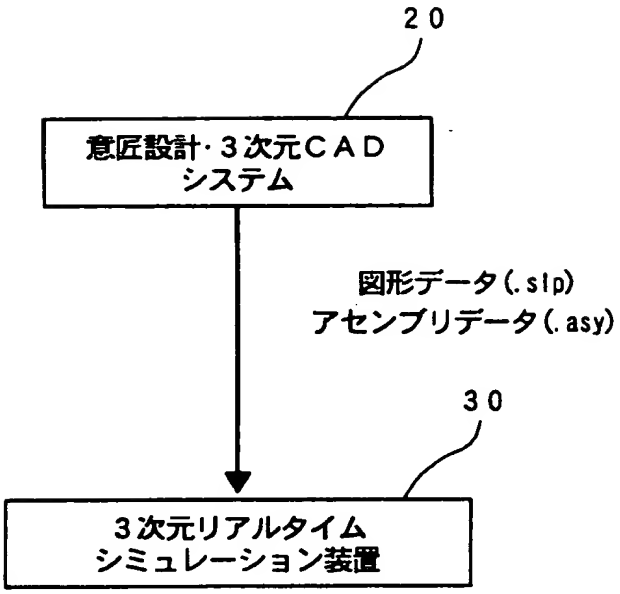
【図３】



【図４】


100
221
CPU
222
RAM
230
ROM
229
モデム
223
ハードディスク
コントローラ
224
フロッピィ
ディスクドライバ
225
CD－ROM
ドライバ
226
マウス
コントローラ
227
キーボード
コントローラ
228
ディスプレイ
コントローラ
211
212
210
104
103
102

【図５】


２０
意匠設計・３次元ＣＡＤ
システム
図形データ(.slp)
アセンブリデータ(.asy)
３０
３次元リアルタイム
シミュレーション装置

【図６】

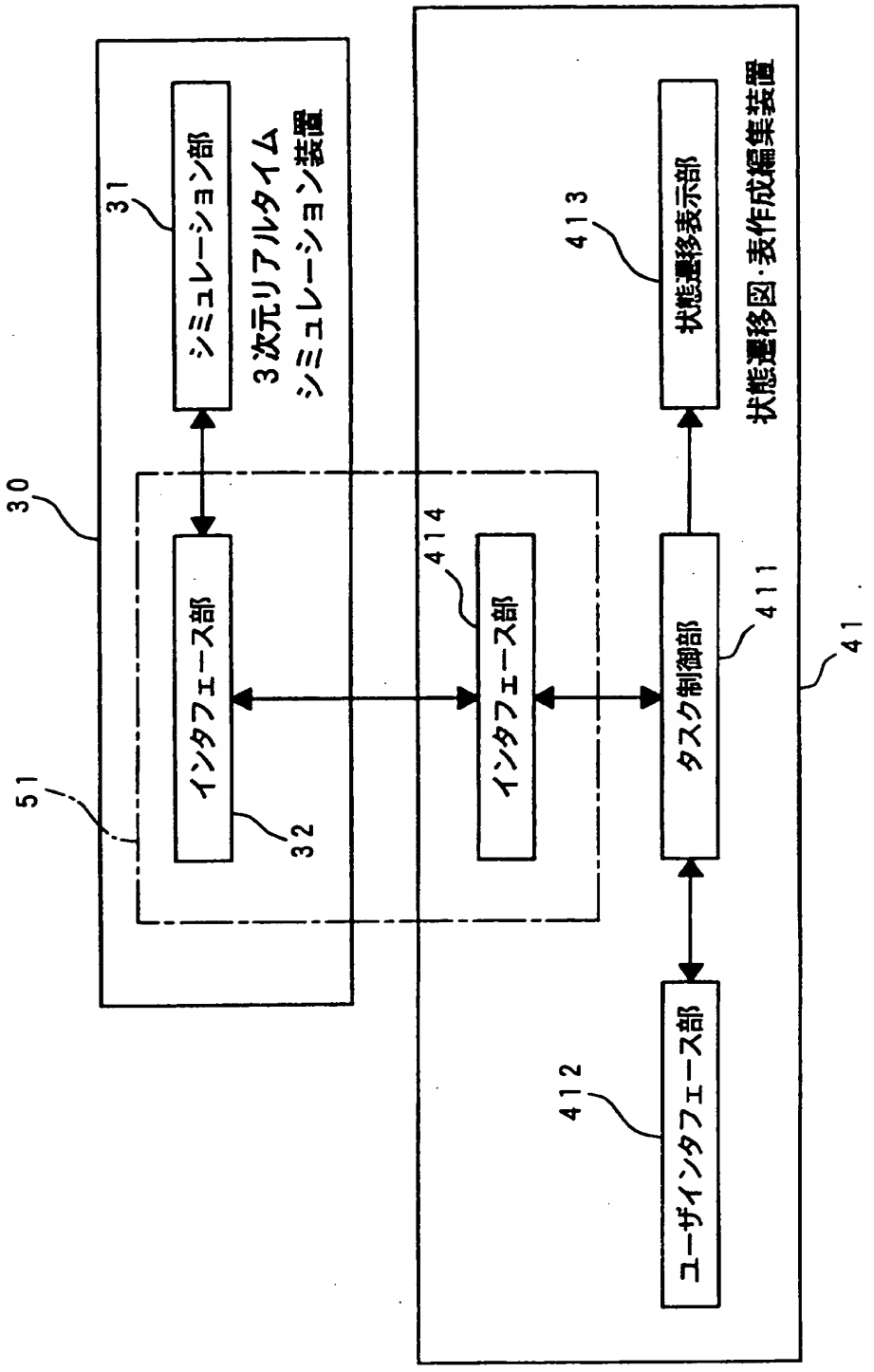
(A)

```
solid Kam_Version2__0_Library1
  color 1.000000 0.660000 0.000000
  facet
      normal 0.000000 0.952236 -0.305364
      normal 0.000000 0.952236 -0.305364
      normal 0.000000 0.952236 -0.305364
    outer loop
      vertex 46.667969 -54.467480 -31.538586
      vertex 46.667969 -57.256718 -40.236450
      vertex 16.667969 -57.256718 -40.236450
    endloop
  endfacet
  facet
      normal 0.000000 0.952236 -0.305364
      normal 0.000000 0.952236 -0.305364
      normal 0.000000 0.952236 -0.305364
    outer loop
      vertex 16.667969 -57.256718 -40.236450
      vertex 16.667969 -54.467480 -31.538586
      vertex 46.667969 -54.467480 -31.538586
    endloop
  endfacet
.
.
.
  facet
      normal 0.000000  -0.472754 0.881194
      normal 0.000000  -0.472754 0.881194
      normal 0.000000  -0.472754 0.881194
    outerloop
      vertex 91.667969 -77.803757 -26.071329
      vertex 91.667969 -73.224937 -23.614828
      vertex 46.667969 -73.224937 -23.614828
    endloop
  endfacet
endsolid Kam_Version2__0_Library1
```

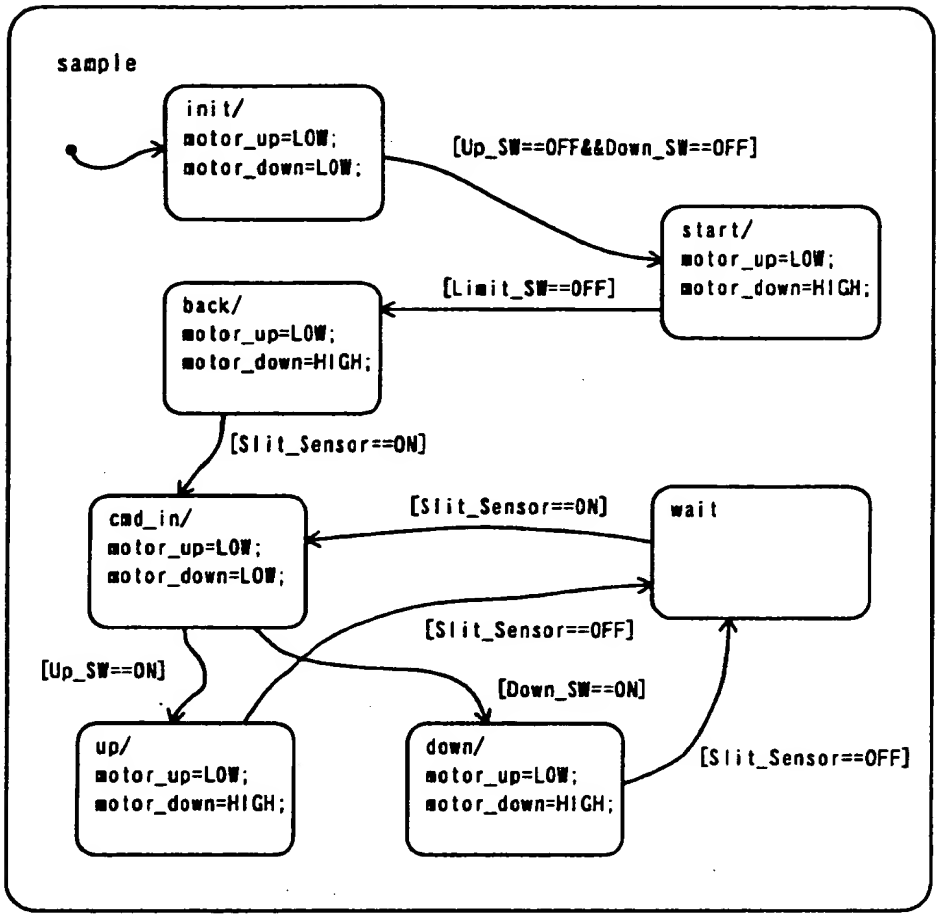
(B)

```
Begin Space
# No 1
 Parent 0 # root
 JointType Free
End

Begin "seat_assy__0_Library1"
 # No 2
 Parent 1 # Space
 RelPosition 0.000000 0.000000 0.000000
 RelAttitude 1.000000 0.000000 0.000000 0.000000
1.000000 0.000000 0.000000 0.000000 1.000000
 JointType Fixed
End

Begin "seat_pan__0_Library1"
 # No 3
 Parent 2 # seat_assy__0_Library1
 Shape seat_pan__0_Library1.slp
 RelPosition 1.810953 51.404652 450.053009
 RelAttitude 1.000000 0.000000 0.000000 0.000000
1.000000 0.000000 0.000000 0.000000 1.000000
 JointType Fixed
End

Begin "Kam_Version2__0_Library1"
 # No 4
 Parent 2 # seat_assy__0_Library1
 Shape Kam_Version2__0_Library1.slp
 RelPosition 201.712204 -211.475098 811.809509
 RelAttitude 0.104385 0.034423 -0.993941 0.994522
-0.009110 0.104131 -0.005471 -0.999366 -0.035185
 JointType Fixed
End
.
.
.
```

【図７】


3次元リアルタイム
シミュレーション装置
シミュレーション部
31
30
インタフェース部
32
51
インタフェース部
414
状態遷移図・表作成編集装置
状態遷移表示部
413
タスク制御部
411
41
ユーザインタフェース部
412

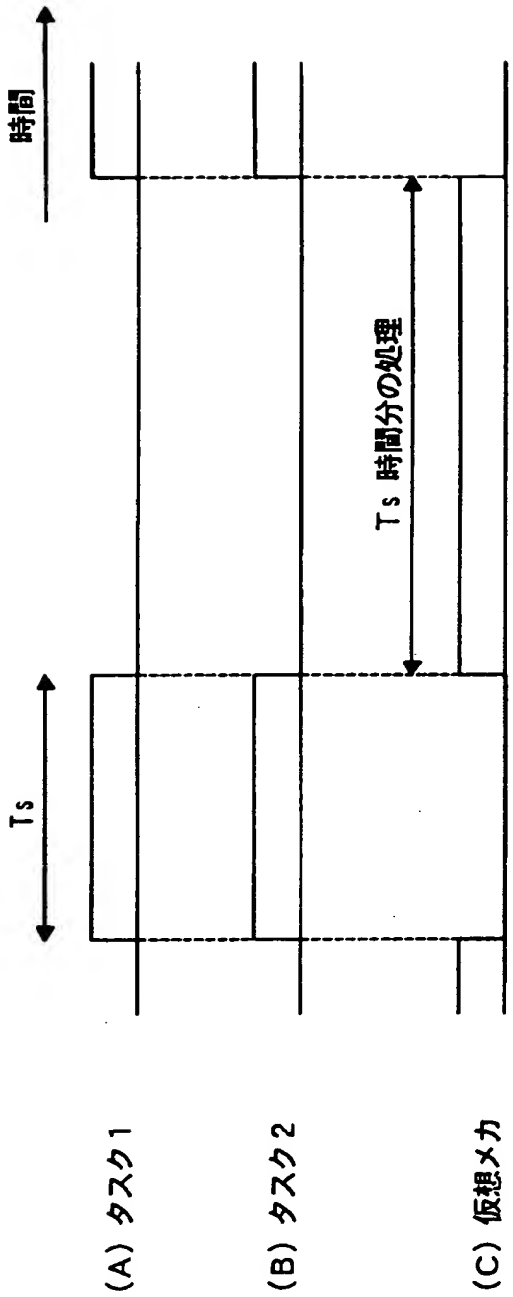
【図８】


sample
init/
motor_up=LOW;
motor_down=LOW;
[Up_SW==OFF&&Down_SW==OFF]
start/
motor_up=LOW;
motor_down=HIGH;
[Limit_SW==OFF]
back/
motor_up=LOW;
motor_down=HIGH;
[Slit_Sensor==ON]
cmd_in/
motor_up=LOW;
motor_down=LOW;
[Slit_Sensor==ON]
wait
[Slit_Sensor==OFF]
[Up_SW==ON]
[Down_SW==ON]
up/
motor_up=LOW;
motor_down=HIGH;
down/
motor_up=LOW;
motor_down=HIGH;
[Slit_Sensor==OFF]

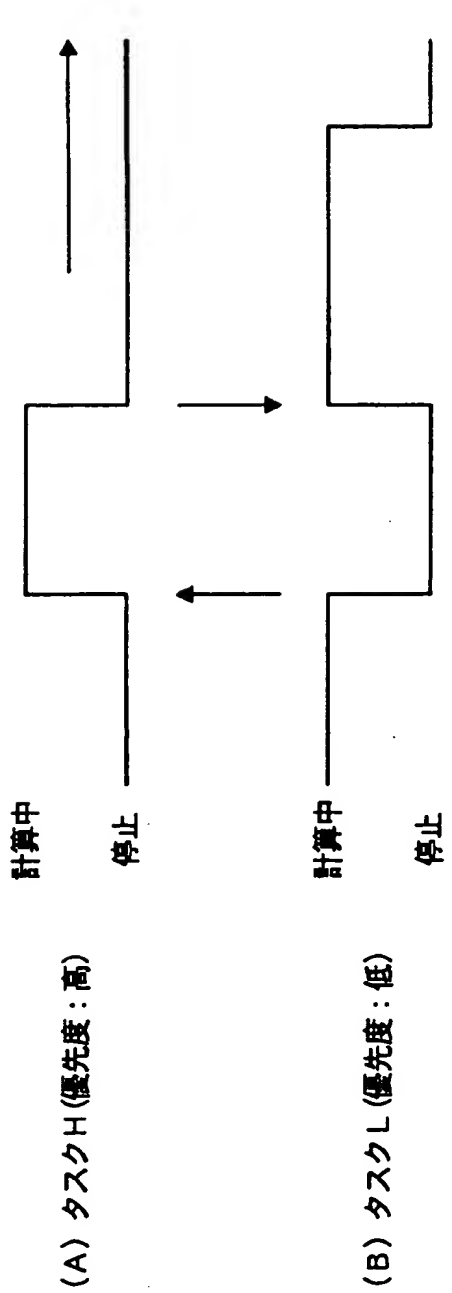
【図９】

	S/E	停止状態	直進中	左ｺｰﾅﾘﾝｸﾞ中	右ｺｰﾅﾘﾝｸﾞ中
		0	1	2	3
信号＿青	0	1 ｱｸｾﾙON（）；	－ ／	－ ／	－ ／
左カーブ	1	－ ／	2 ｱｸｾﾙOFF（）； 左旋回（）；	－ 左旋回（）；	2 左旋回（）；
右カーブ	2	－ ／	3 ｱｸｾﾙOFF（）； 右旋回（）；	3 右旋回（）；	3 右旋回（）；
直線	3	－ ／	－ ／	1 ﾊﾝﾄﾞﾙを戻す（）； ｱｸｾﾙON（）；	1 ﾊﾝﾄﾞﾙを戻す（）； ｱｸｾﾙON（）；

【図１０】

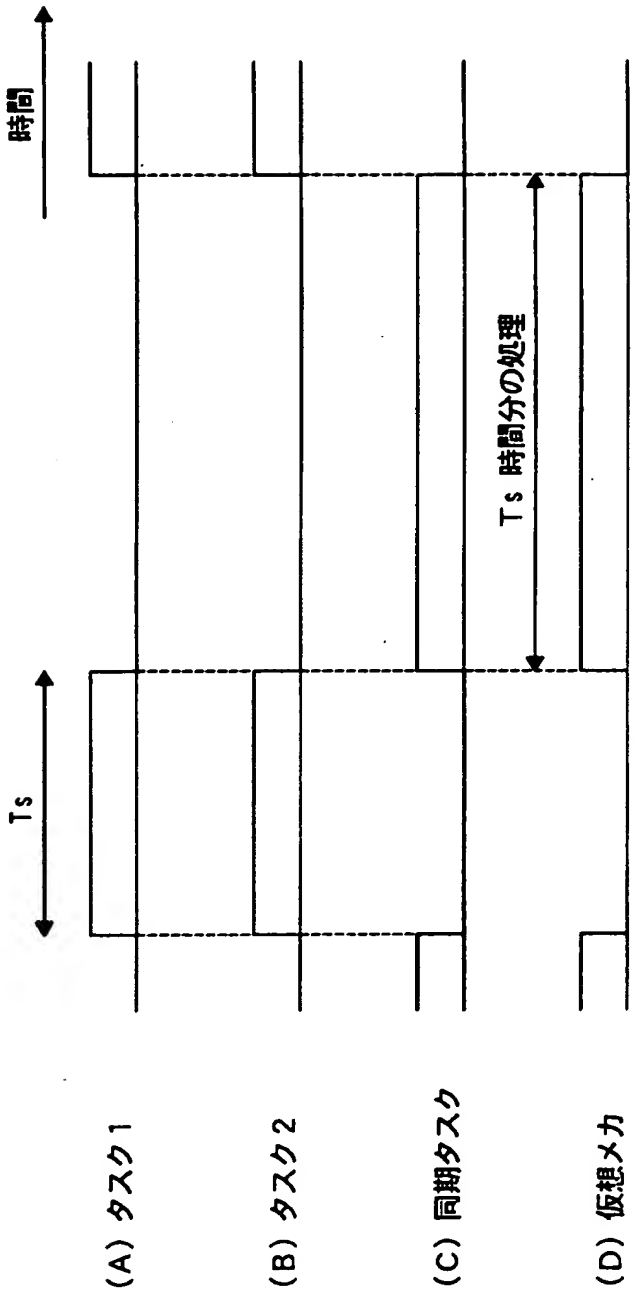


【図１１】

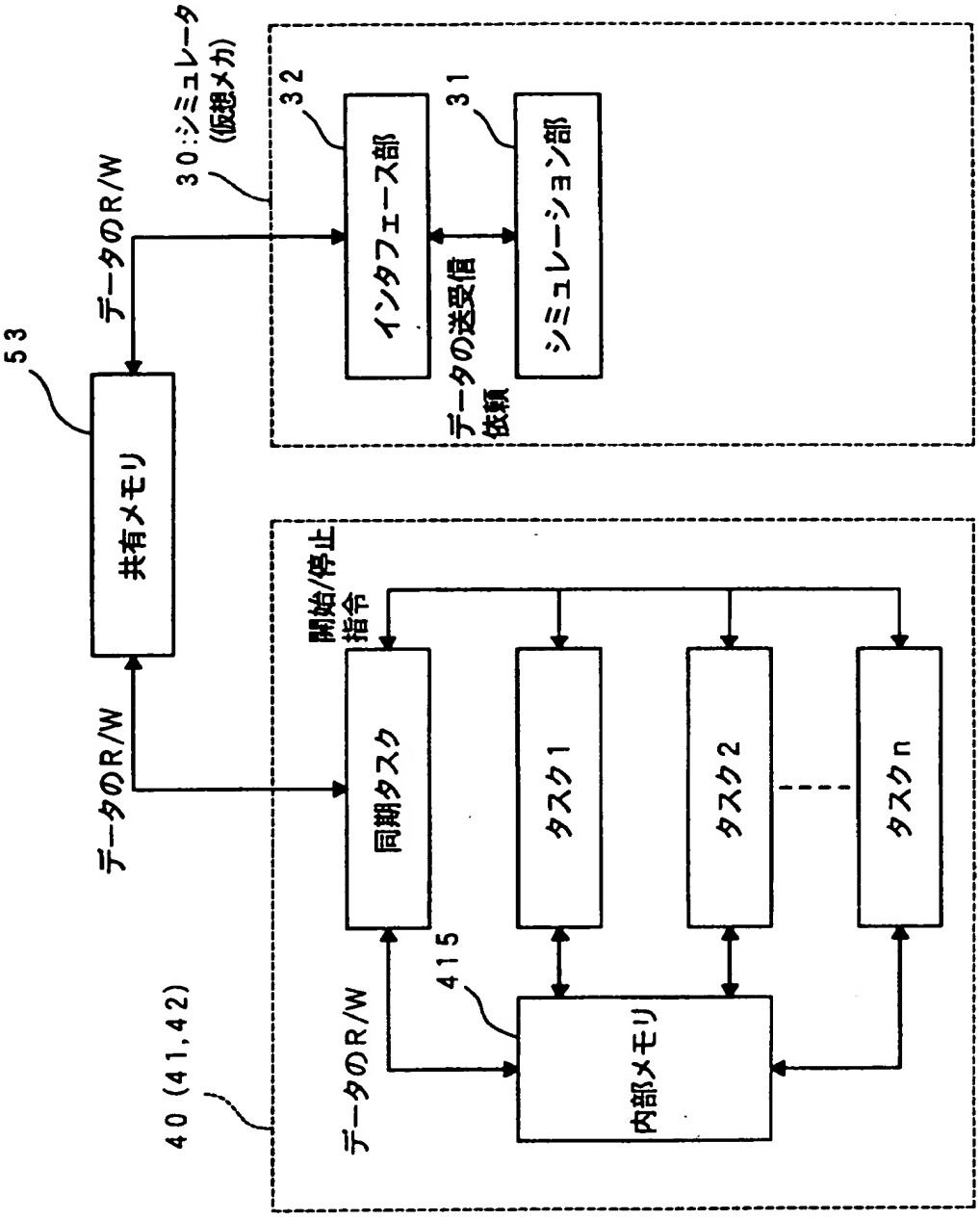


出証特２０００－３０８０２４８

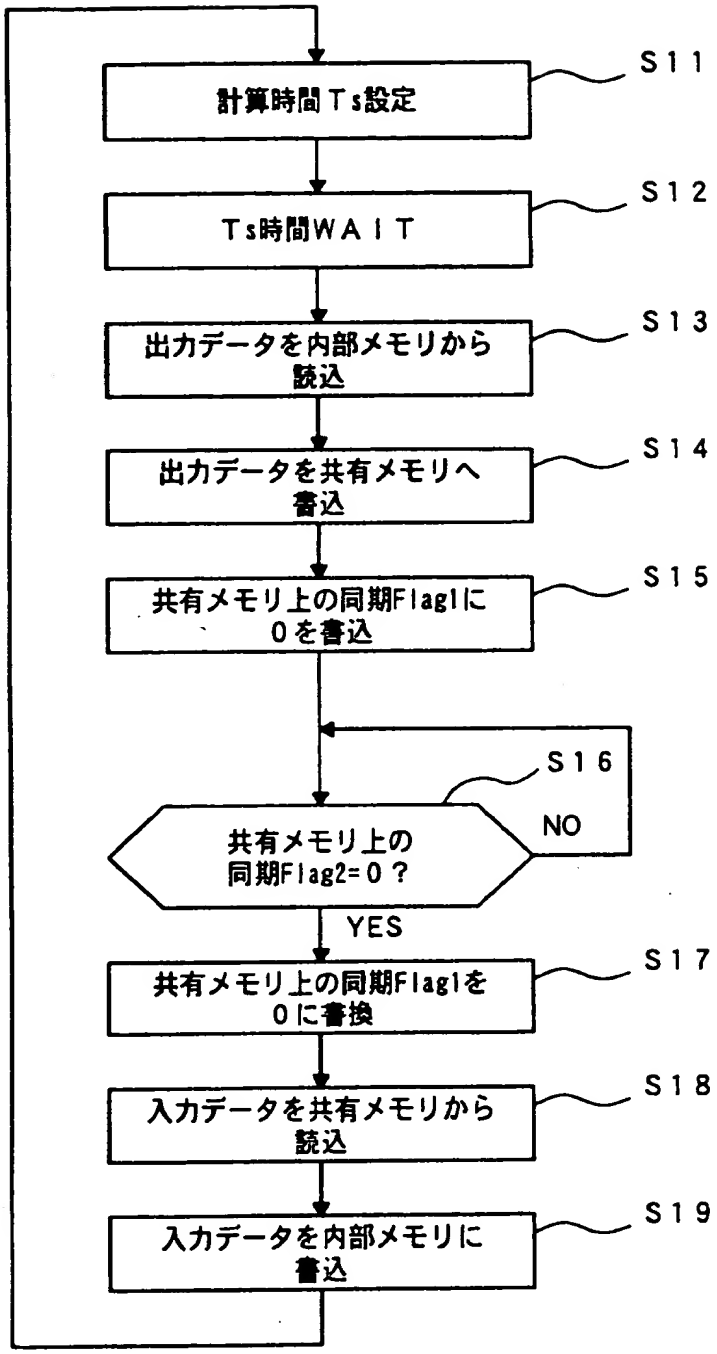
【図１２】



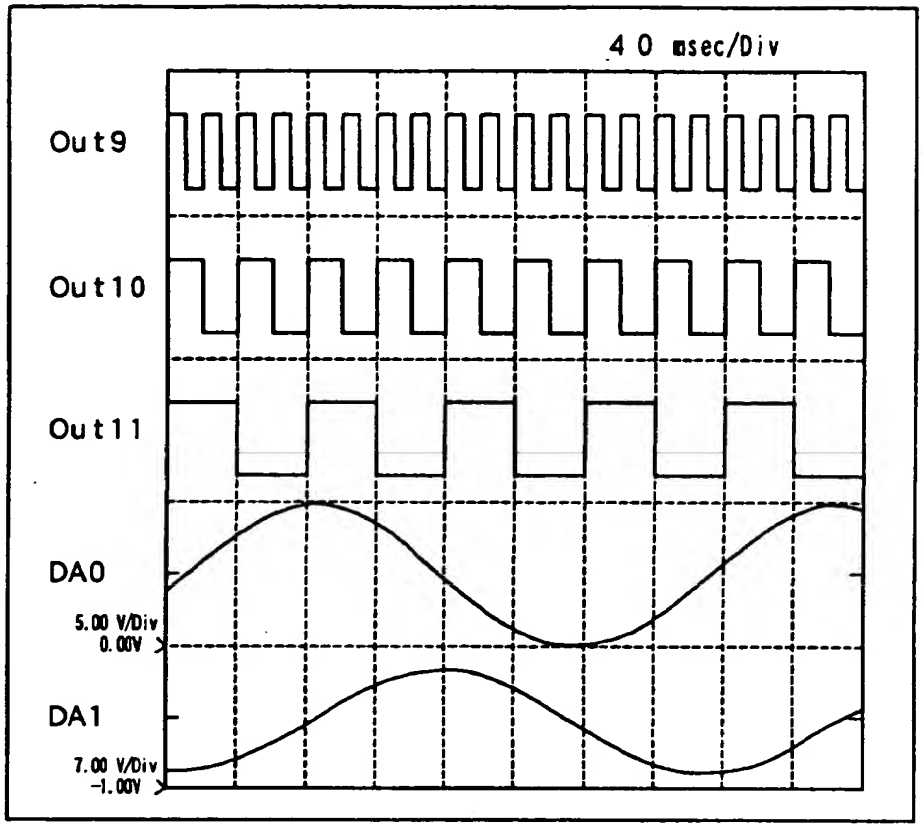
【図１３】


30:シミュレータ
(仮想メカ)
32
31
インタフェース部
シミュレーション部
データのR/W
データの送受信
依頼
53
共有メモリ
開始/停止
指令
同期タスク
タスク1
タスク2
タスクn
データのR/W
415
内部メモリ
40(41,42)

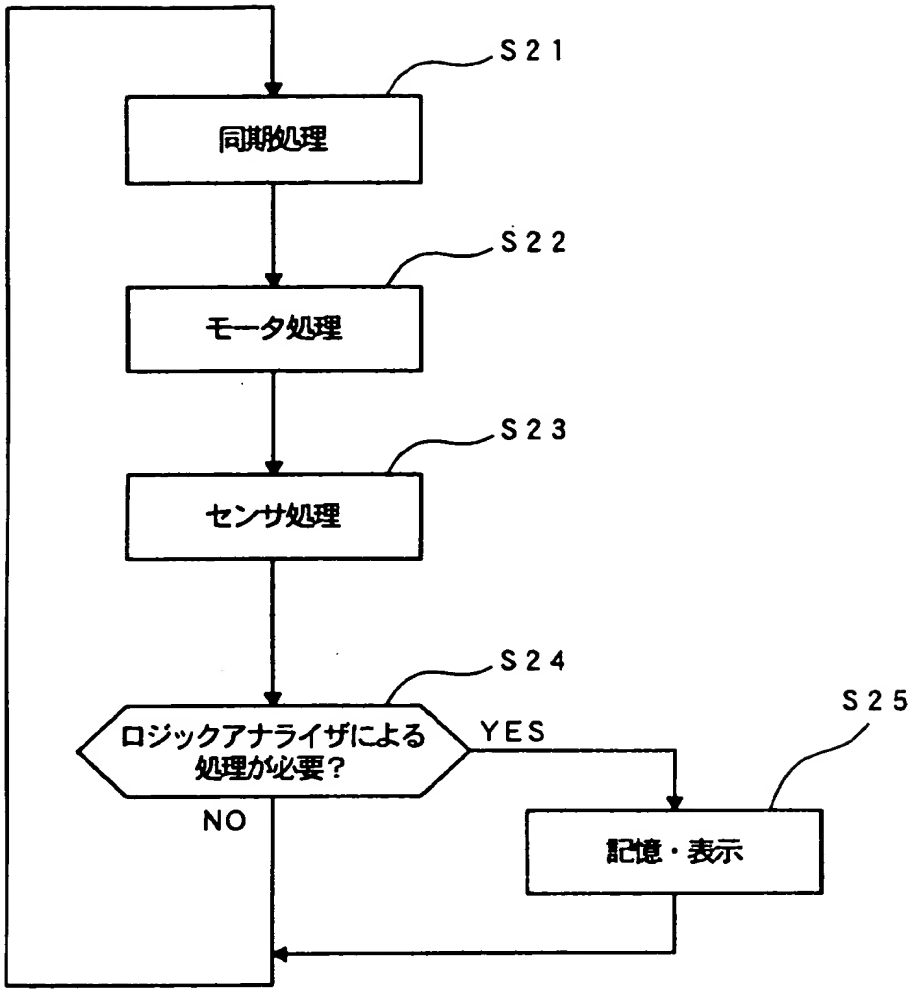
【図１４】


S11
計算時間Ts設定
S12
Ts時間WAIT
S13
出力データを内部メモリから読込
S14
出力データを共有メモリへ書込
S15
共有メモリ上の同期Flag1に0を書込
S16
NO
共有メモリ上の同期Flag2=0？
YES
S17
共有メモリ上の同期Flag1を0に書換
S18
入力データを共有メモリから読込
S19
入力データを内部メモリに書込

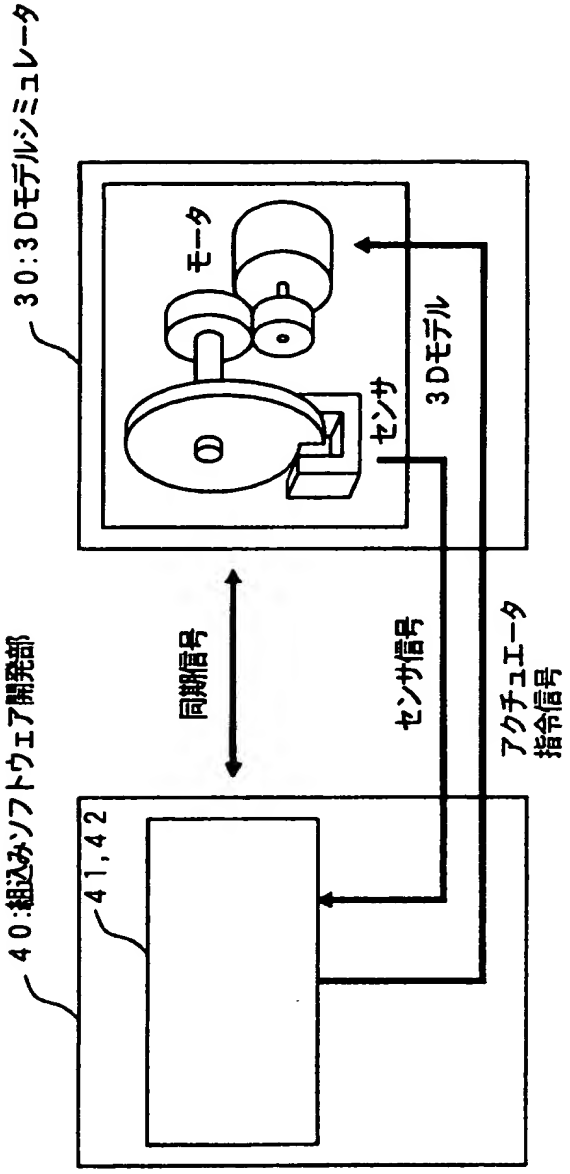
【図１５】


40 msec/Div
Out9
Out10
Out11
DA0
5.00 V/Div
0.00V
DA1
7.00 V/Div
-1.00V

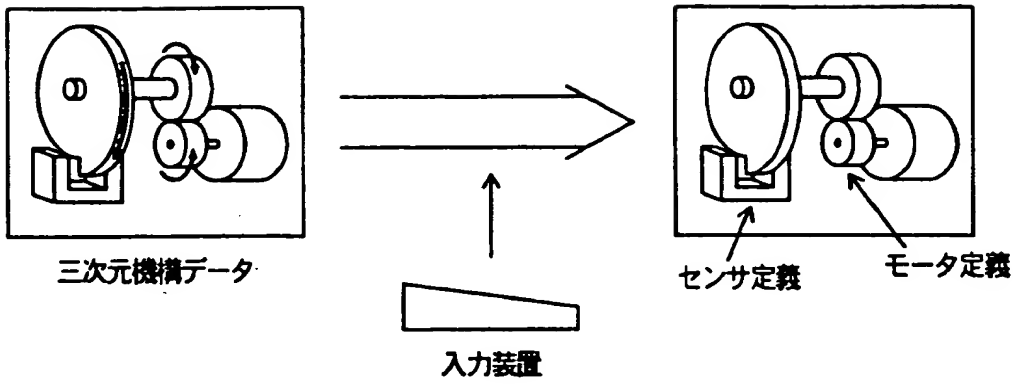
【図１６】


S21
同期処理
S22
モータ処理
S23
センサ処理
S24
ロジックアナライザによる
処理が必要？
YES
NO
S25
記憶・表示

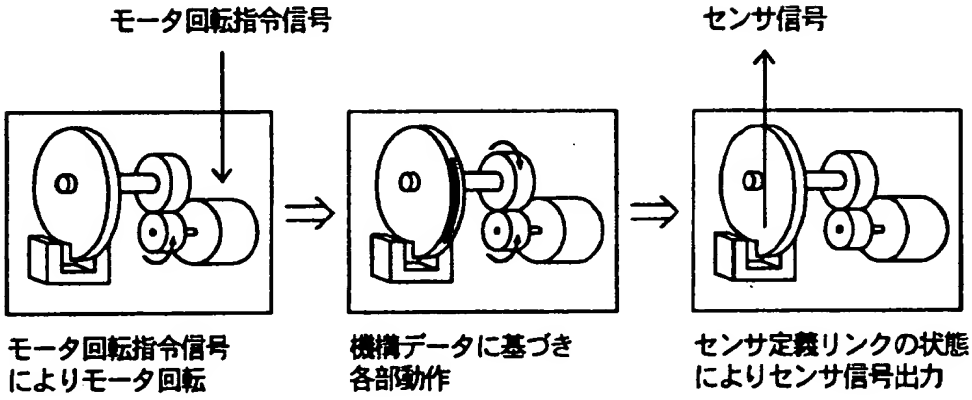
【図１７】


３０:３Dモデルシミュレータ
モータ
センサ
３Dモデル
同期信号
センサ信号
アクチュエータ
指令信号
４０:組込みソフトウェア開発部
４１,４２

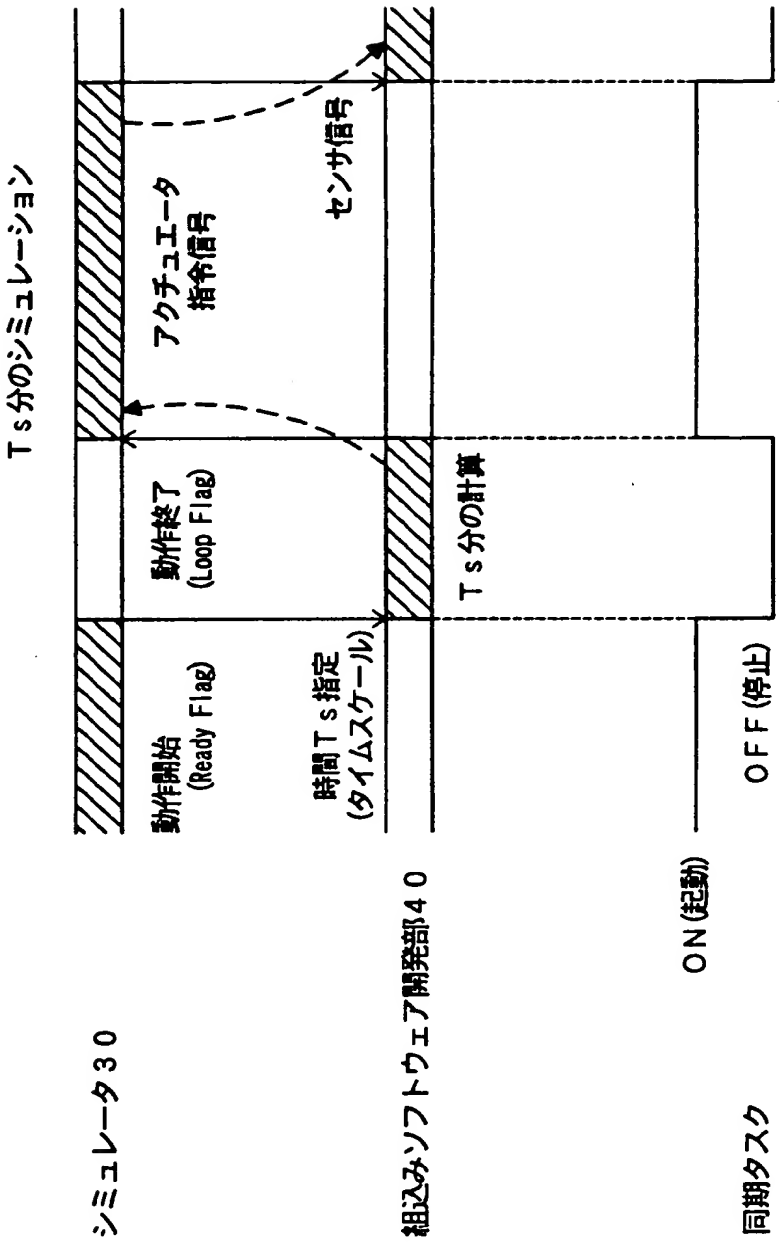
【図１８】


三次元機構データ
センサ定義
モータ定義
入力装置

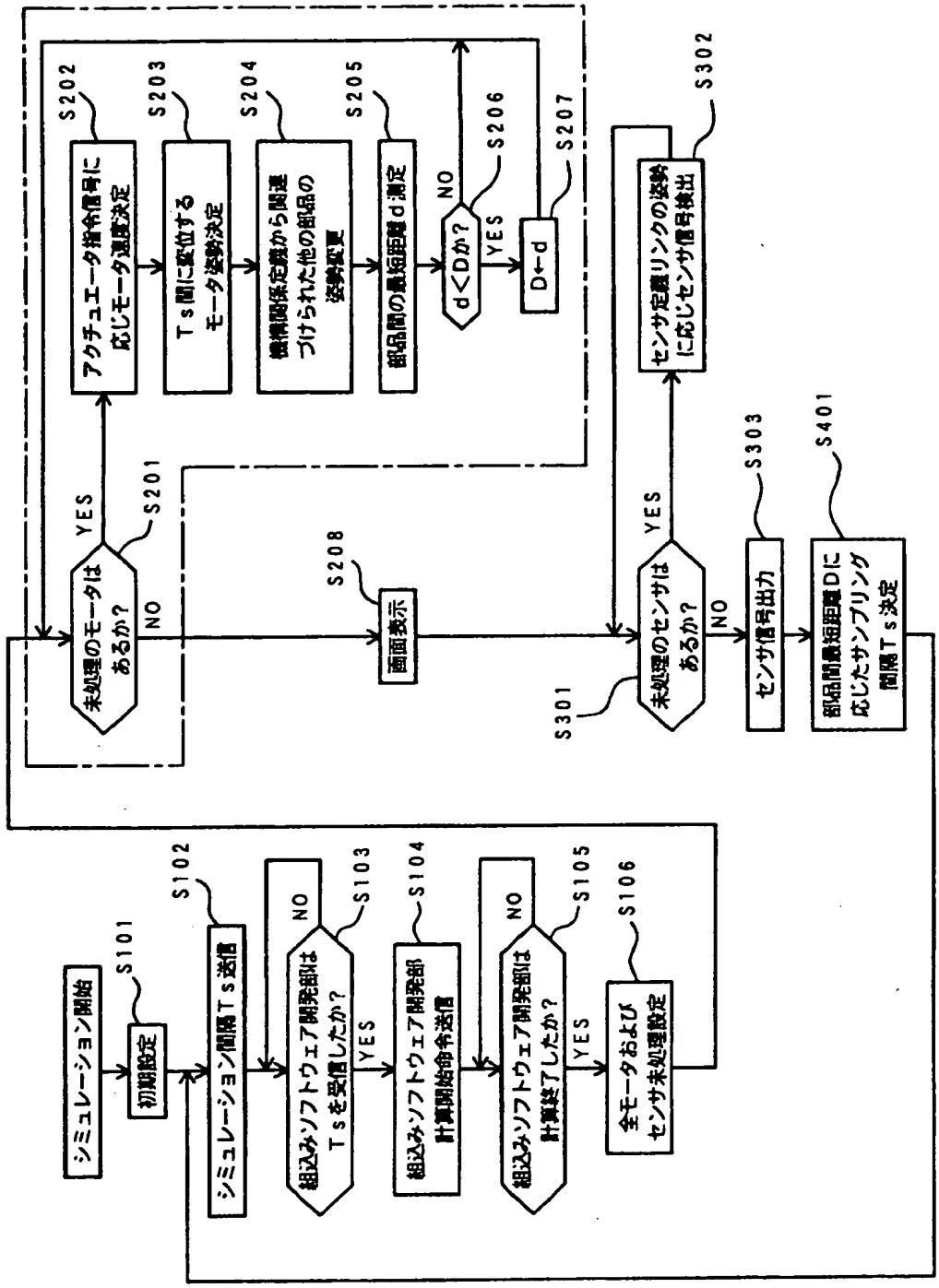
【図１９】


モータ回転指令信号
センサ信号
モータ回転指令信号
によりモータ回転
機構データに基づき
各部動作
センサ定義リンクの状態
によりセンサ信号出力

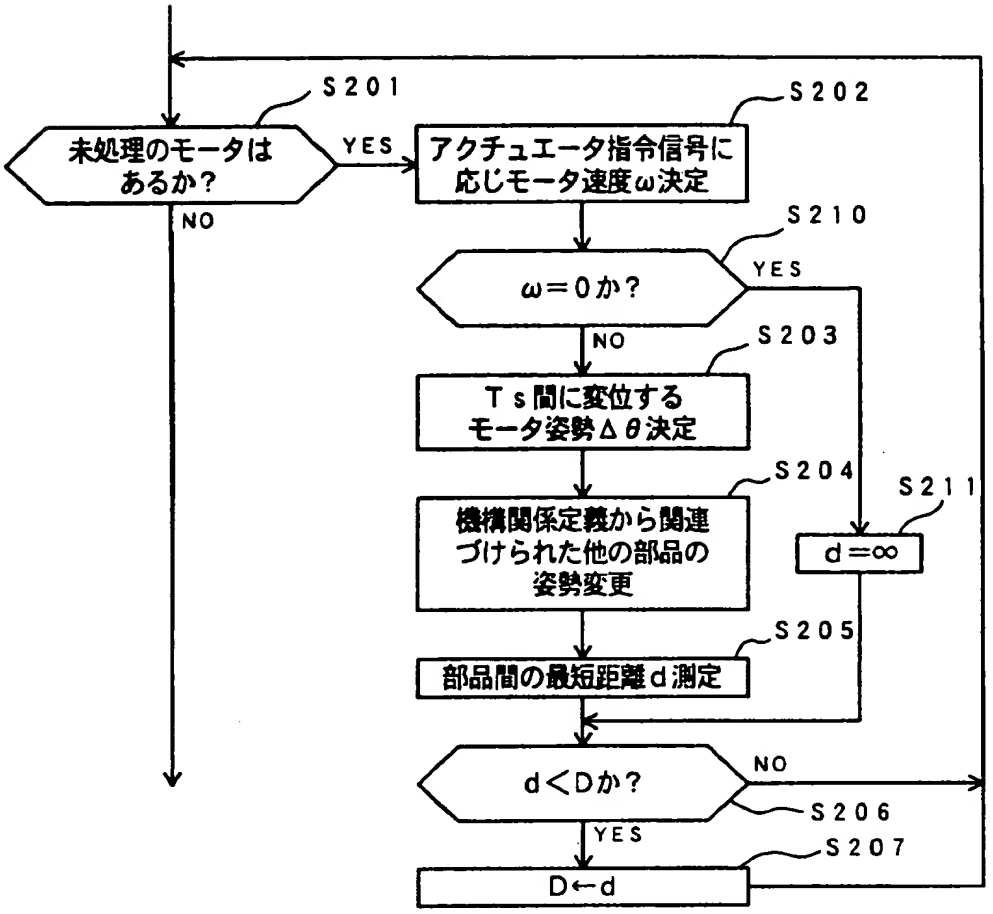
【図２０】


Ｔｓ分のシミュレーション
センサ信号
アクチュエータ
指令信号
動作終了
(Loop Flag)
Ｔｓ分の計算
動作開始
(Ready Flag)
時間Ｔｓ指定
(タイムスケール)
ＯＦＦ(停止)
ＯＮ(起動)
シミュレータ３０
組込みソフトウェア開発部４０
同期タスク

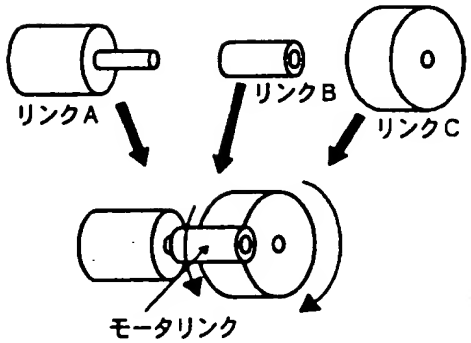
【図２１】


シミュレーション開始
S101
初期設定
S102
シミュレーション間隔Ｔｓ進捗
S103
組込みソフトウェア開発部はＴｓを受信したか？
NO
YES
S104
組込みソフトウェア開発部計算開始命令送信
S105
組込みソフトウェア開発部は計算終了したか？
NO
YES
S106
全モータおよびセンサ未処理設定
S201
未処理のモータはあるか？
YES
NO
S202
アクチュエータ指令信号に応じモータ速度決定
S203
Ｔｓ間に変位するモータ姿勢決定
S204
機構関係定義から関連づけられた他の部品の姿勢変更
S205
部品間の最短距離ｄ測定
S206
d＜Dか？
NO
YES
S207
D←d
S208
画面表示
S301
未処理のセンサはあるか？
YES
NO
S302
センサ定義リンクの姿勢に応じセンサ信号検出
S303
センサ信号出力
S401
部品間最短距離Dに応じたサンプリング間隔Ｔｓ決定

【図２２】

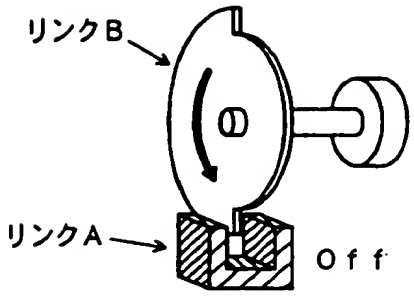

S201
未処理のモータは
あるか？
YES
NO
S202
アクチュエータ指令信号に
応じモータ速度ω決定
S210
ω＝０か？
YES
NO
S203
Ｔｓ間に変位する
モータ姿勢Δθ決定
S204
機構関係定義から関連
づけられた他の部品の
姿勢変更
S211
ｄ＝∞
S205
部品間の最短距離ｄ測定
ｄ＜Ｄか？
NO
S206
YES
S207
D←d

【図２３】

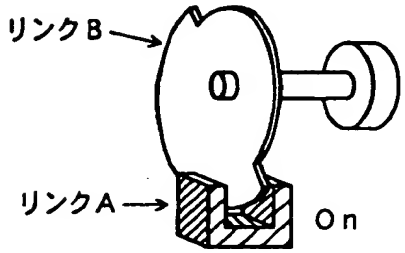

リンクA
リンクB
リンクC
モータリンク

【図２４】

(A)

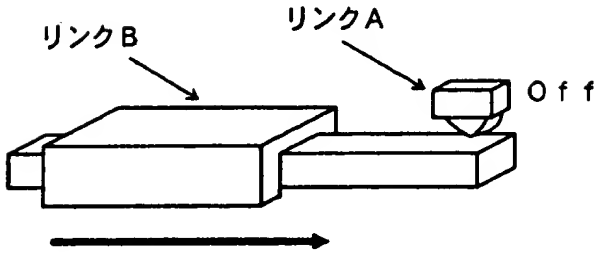


(B)

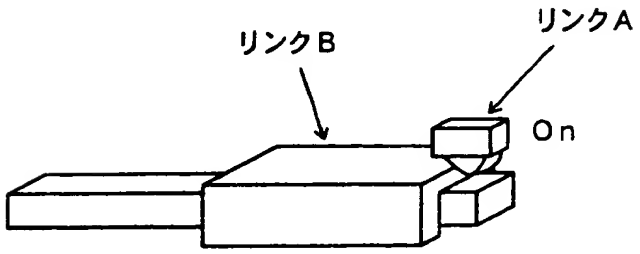


【図２５】

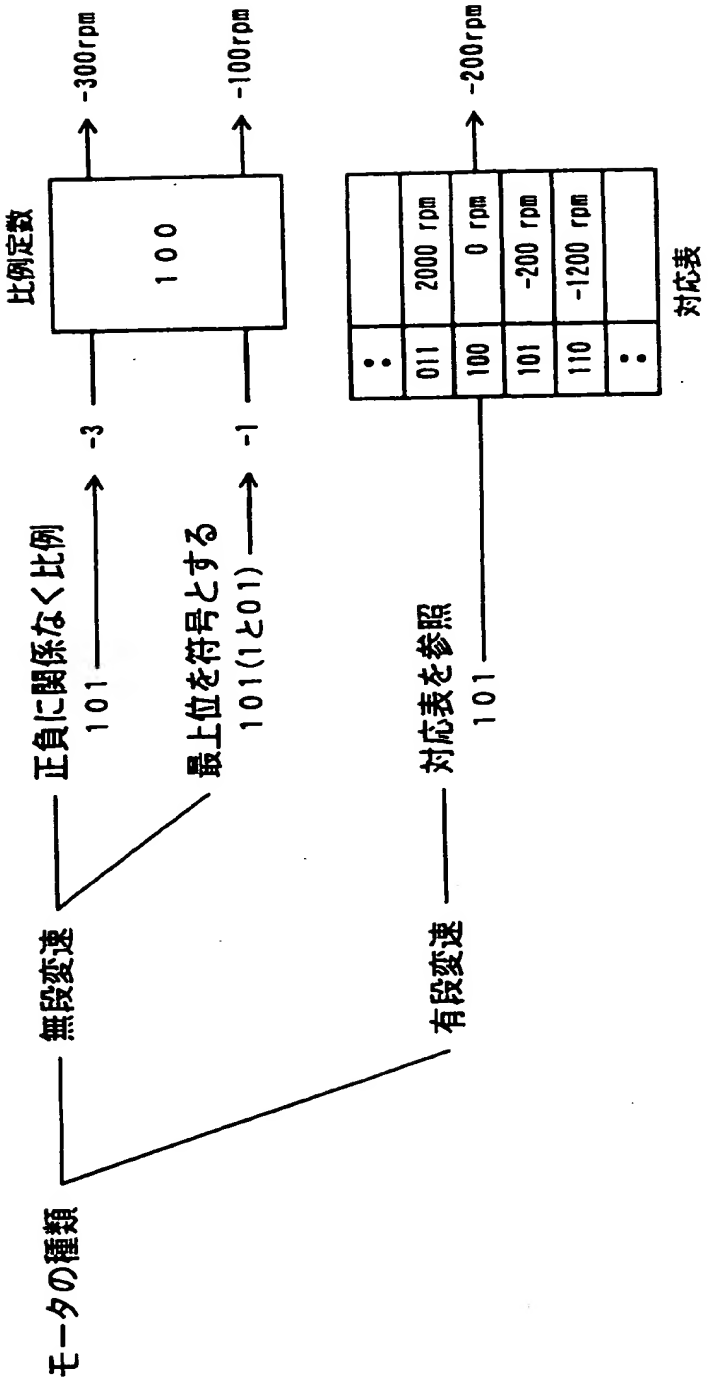
(A)



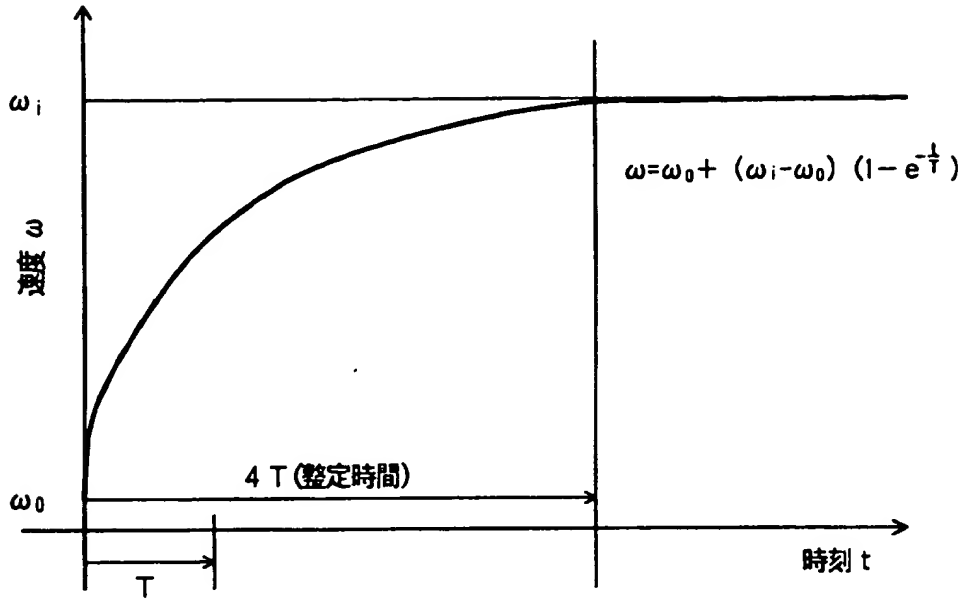
(B)



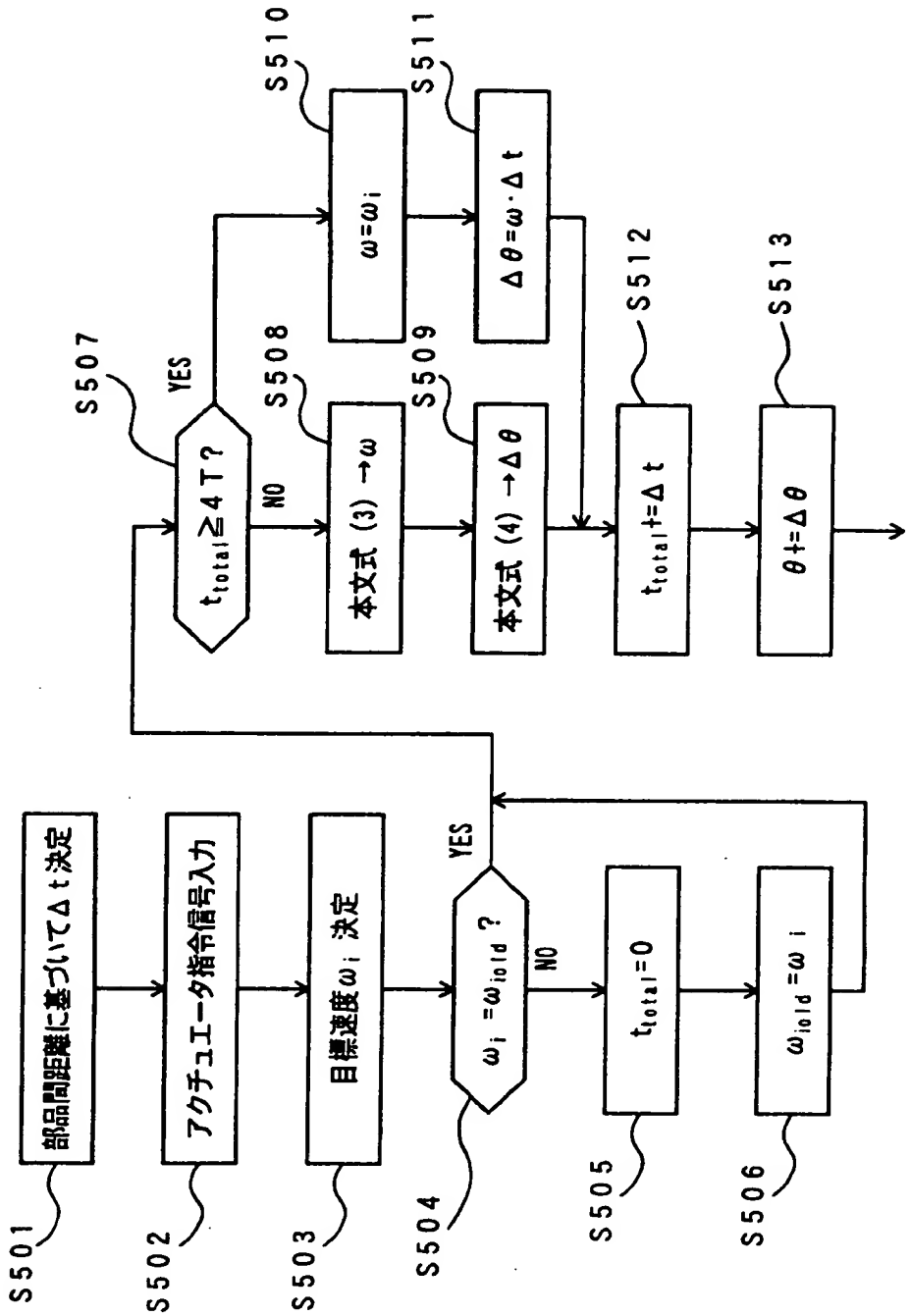
【図２６】


モータの種類
無段変速
有段変速
正負に関係なく比例
101
最上位を符号とする
101(1と01)
対応表を参照
101
-3
-1
比例定数
100
-300rpm
-100rpm
-200rpm
対応表
011
100
101
110
2000 rpm
0 rpm
-200 rpm
-1200 rpm

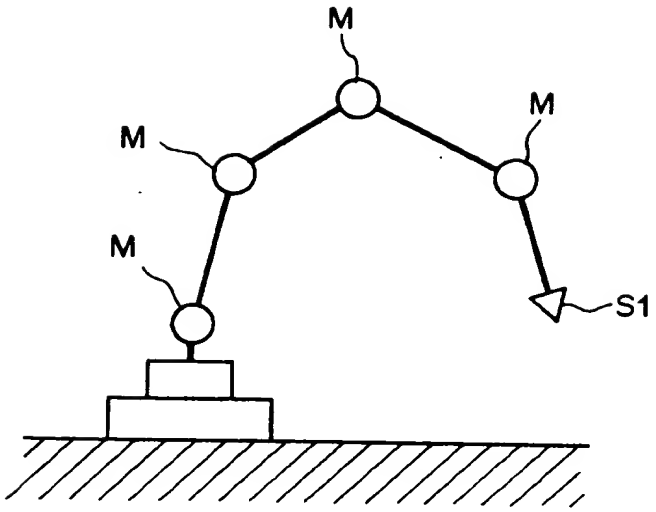
【図２７】



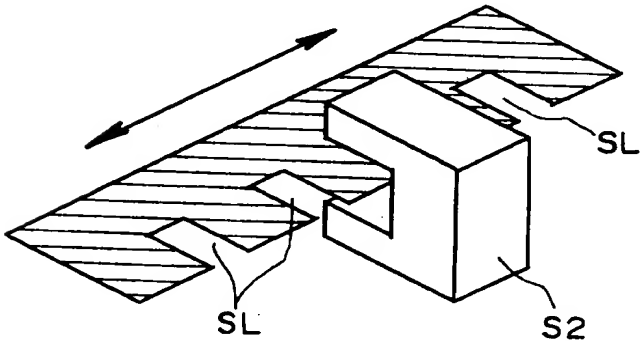
【図２８】


S501
部品間距離に基づいてΔt決定
S502
アクチュエータ指令信号入力
S503
目標速度ωi 決定
S504
ωi =ωiold ?
YES
NO
S505
ttotal=0
S506
ωiold =ωi
S507
ttotal≧4T?
YES
NO
S508
本文式 (3) →ω
S509
本文式 (4) →Δθ
S510
ω=ωi
S511
Δθ=ω·Δt
S512
ttotal+=Δt
S513
θ+=Δθ

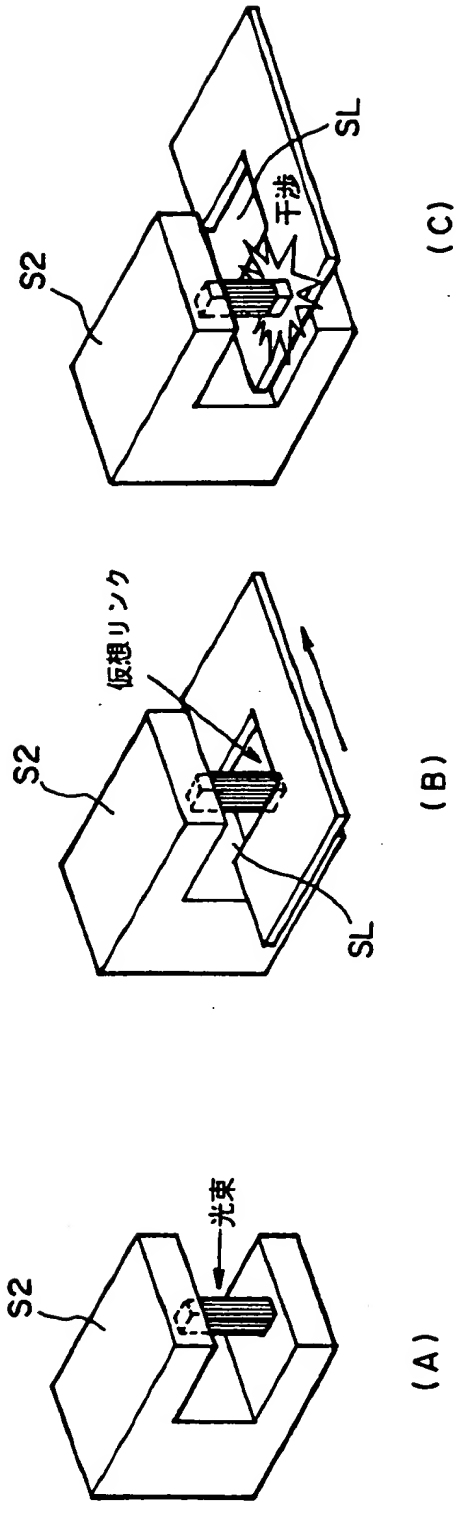
【図２９】



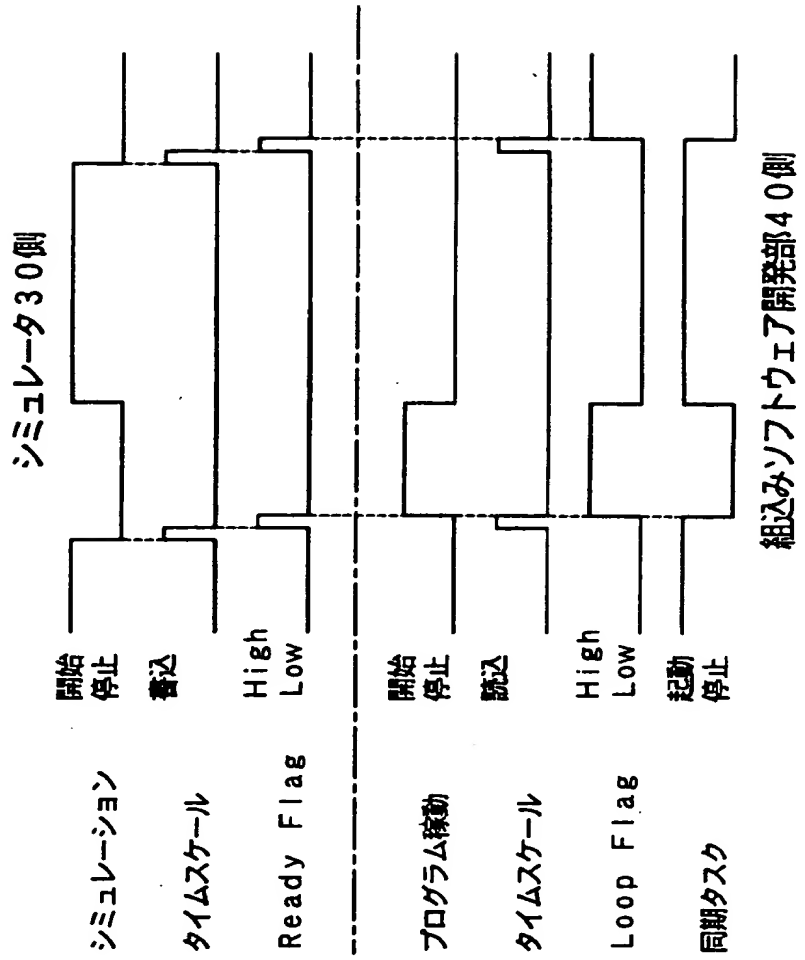
【図３０】



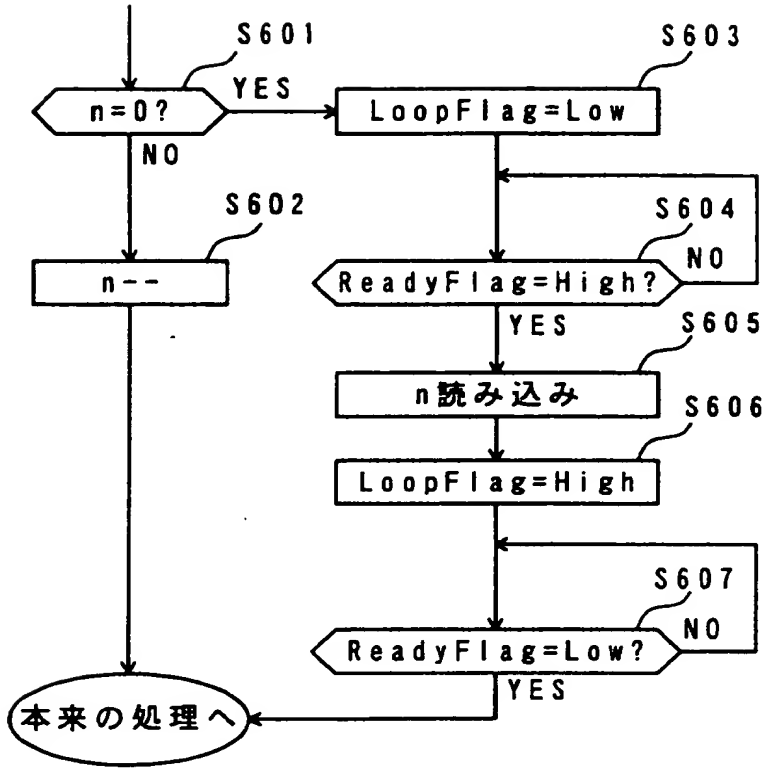
【図３１】



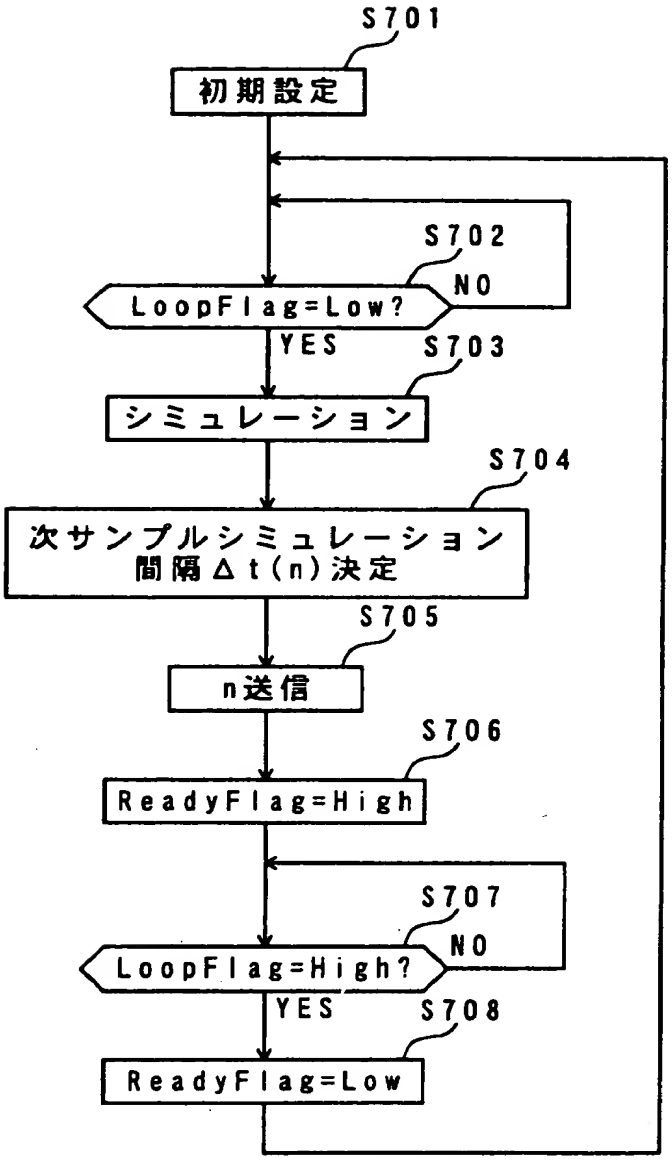
【図３２】


シミュレータ３０側
組込みソフトウェア開発部４０側
シミュレーション
タイムスケール
Ready Flag
プログラム稼動
タイムスケール
Loop Flag
同期タスク
開始
停止
書込
High
Low
開始
停止
読込
High
Low
起動
停止

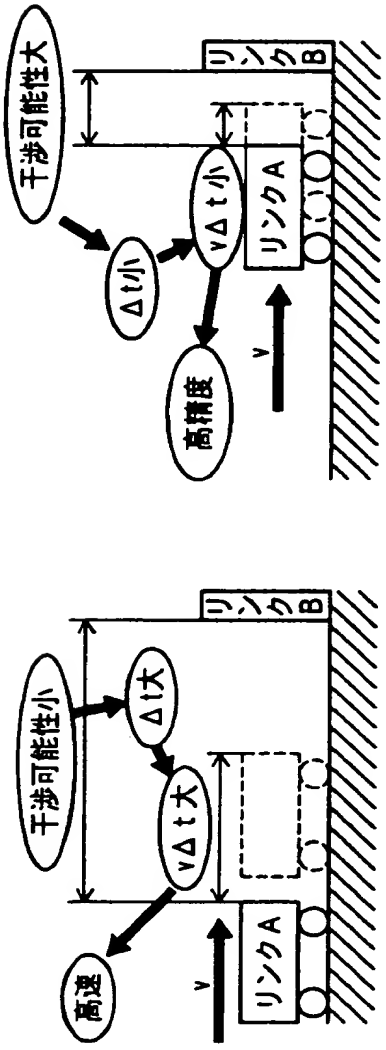
【図３３】


S601
n=0?
YES
NO
S602
n--
S603
LoopFlag=Low
S604
ReadyFlag=High?
NO
YES
S605
n読み込み
S606
LoopFlag=High
S607
ReadyFlag=Low?
NO
YES
本来の処理へ

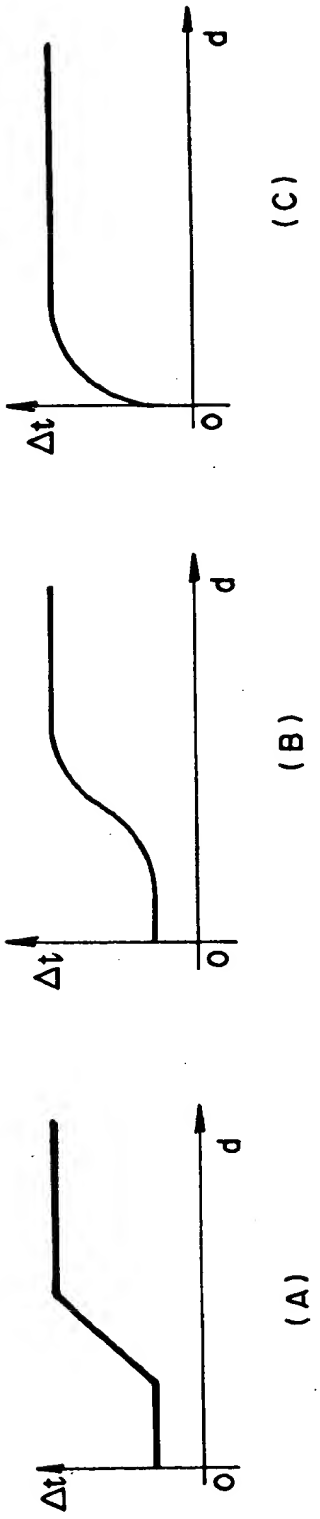
【図３４】


S701
初期設定
S702
NO
LoopFlag=Low?
YES
S703
シミュレーション
S704
次サンプルシミュレーション
間隔Δt(n)決定
S705
n送信
S706
ReadyFlag=High
S707
NO
LoopFlag=High?
YES
S708
ReadyFlag=Low

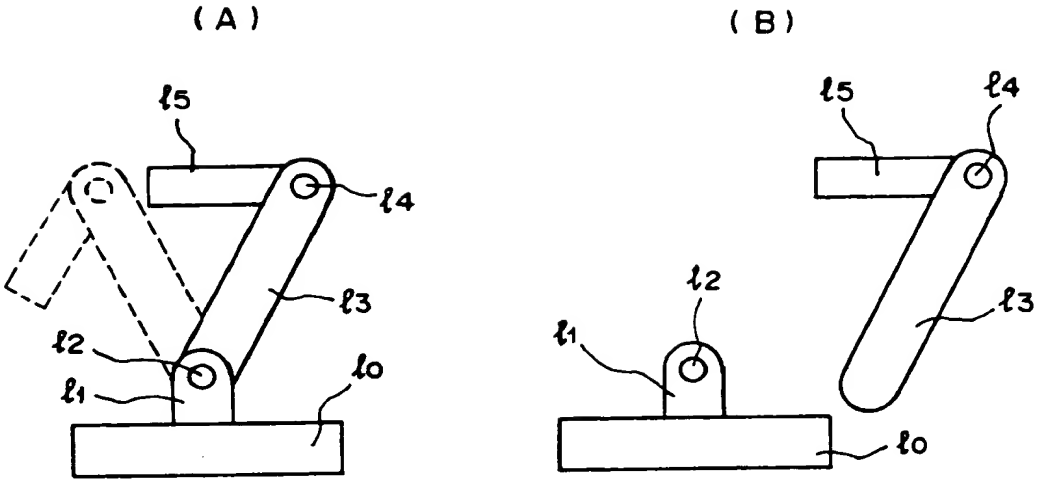
【図３５】


干渉可能性大
リンクB
vΔt小
リンクA
Δt小
高精度
v
(B)
干渉可能性小
リンクB
Δt大
vΔt大
リンクA
高速
v
(A)

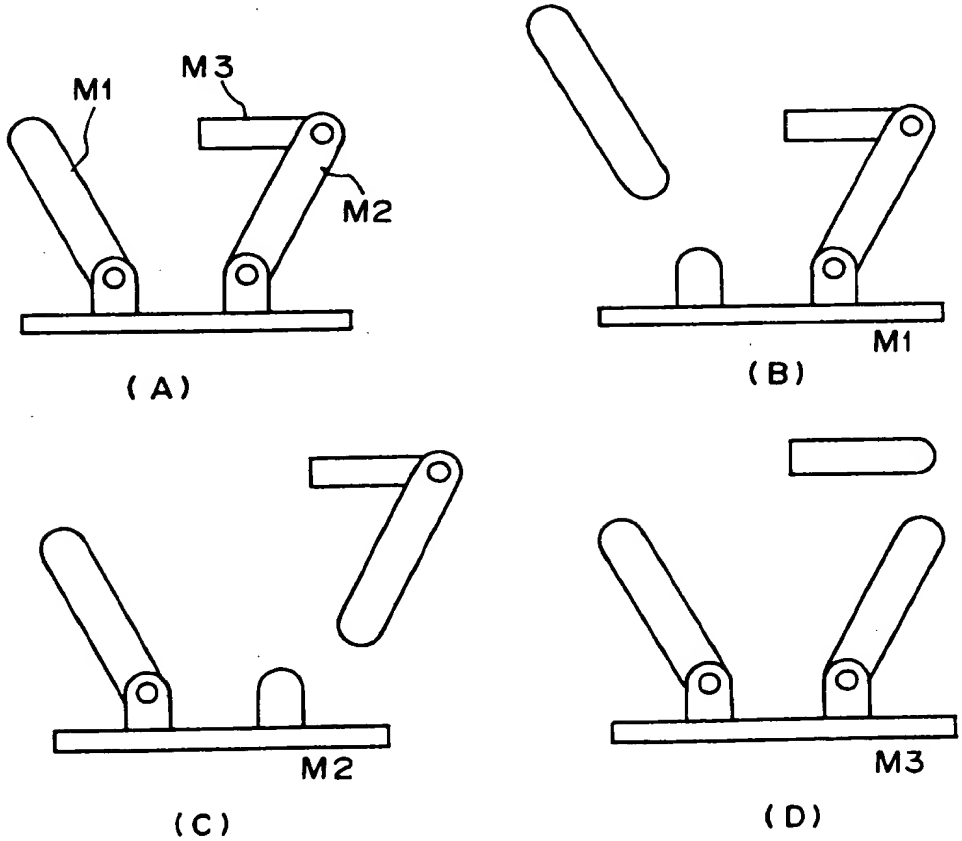
【図３６】



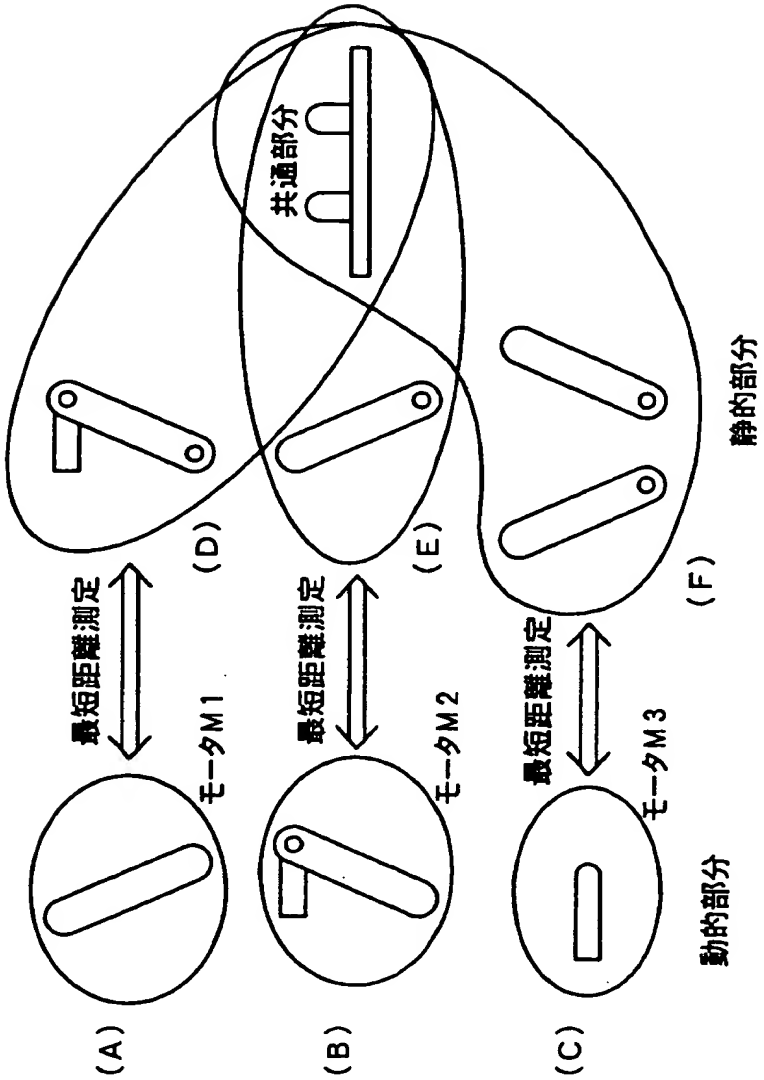
【図３７】



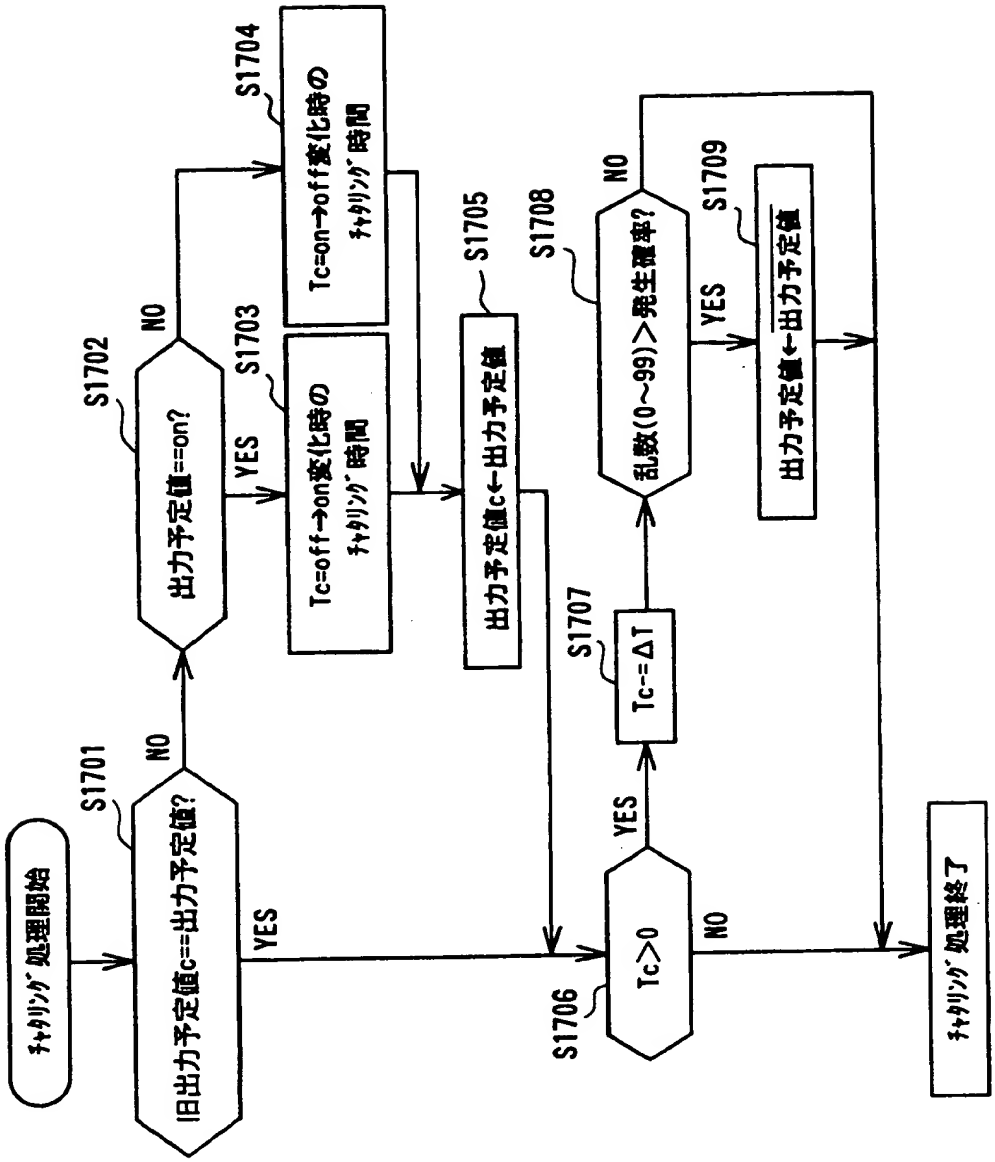
【図３８】



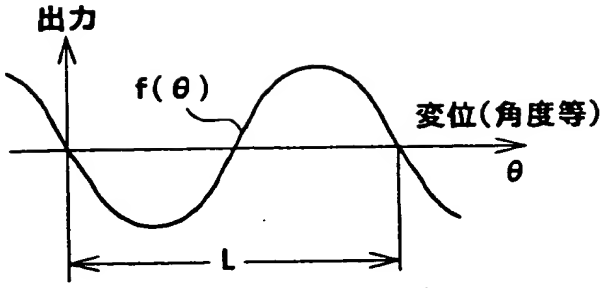
【図３９】


共通部分
静的部分
動的部分
最短距離測定
モータM1
最短距離測定
モータM2
最短距離測定
モータM3
(A)
(B)
(C)
(D)
(E)
(F)

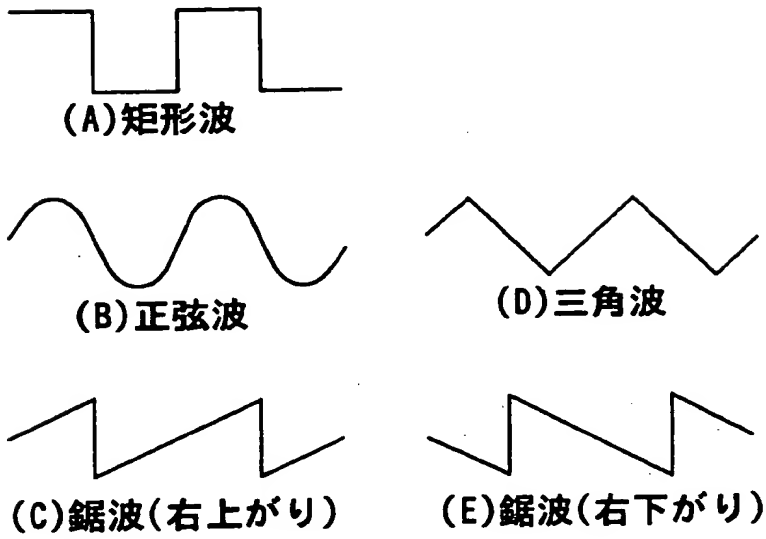
【図５２】


チャタリング処理開始
S1701
旧出力予定値c==出力予定値?
NO
YES
S1702
出力予定値==on?
NO
YES
S1703
Tc=off→on変化時のチャタリング時間
S1704
Tc=on→off変化時のチャタリング時間
S1705
出力予定値c←出力予定値
S1706
Tc>0
YES
NO
S1707
Tc-=ΔT
S1708
乱数(0～99)>発生確率?
NO
YES
S1709
出力予定値←出力予定値
チャタリング処理終了

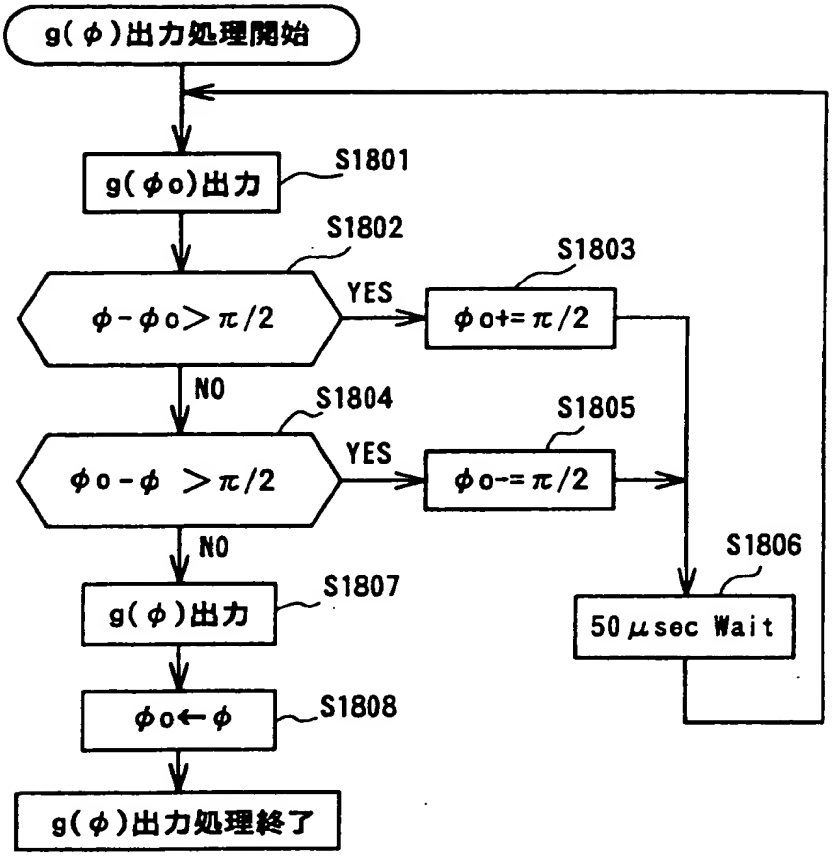
【図５３】



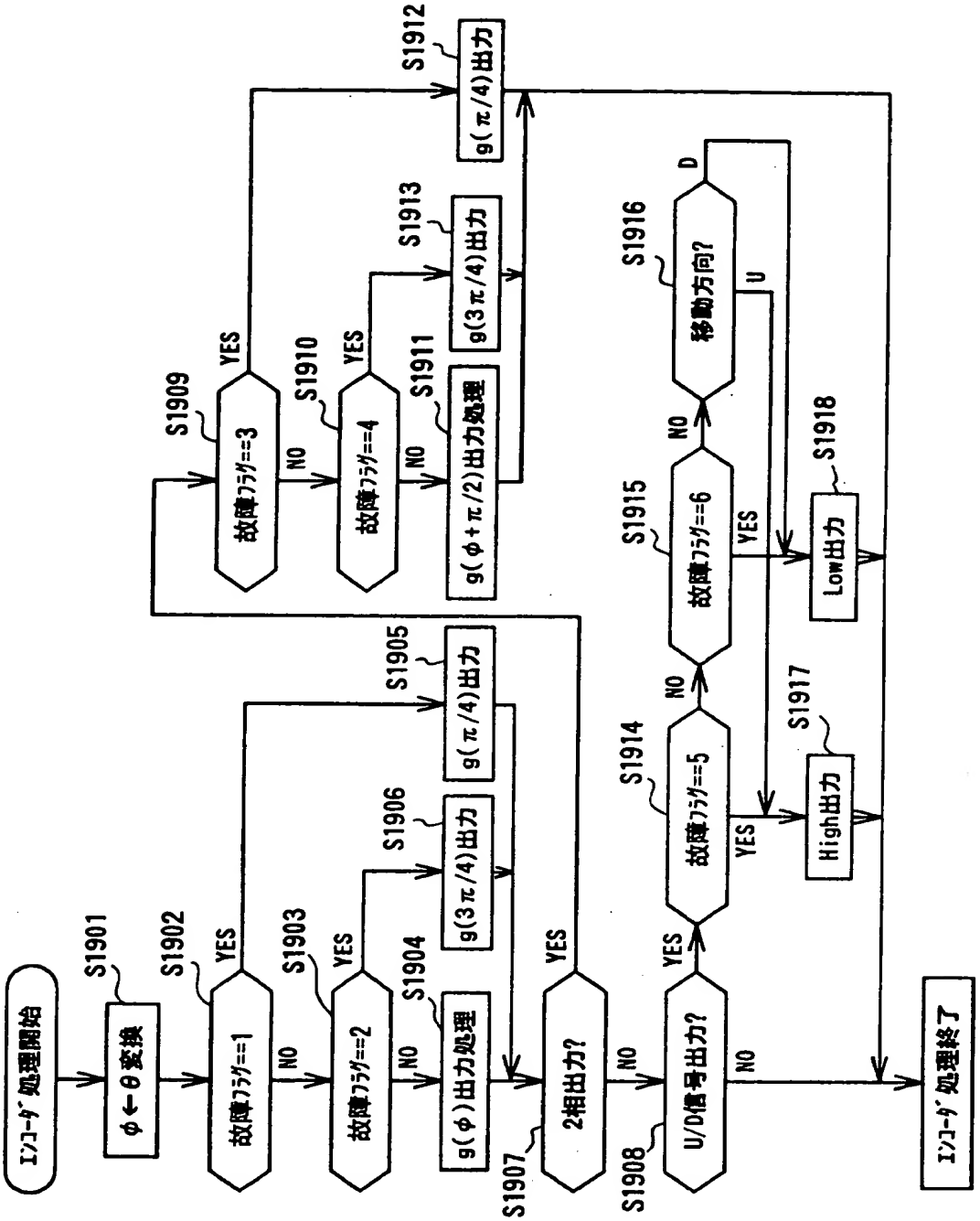
【図５４】



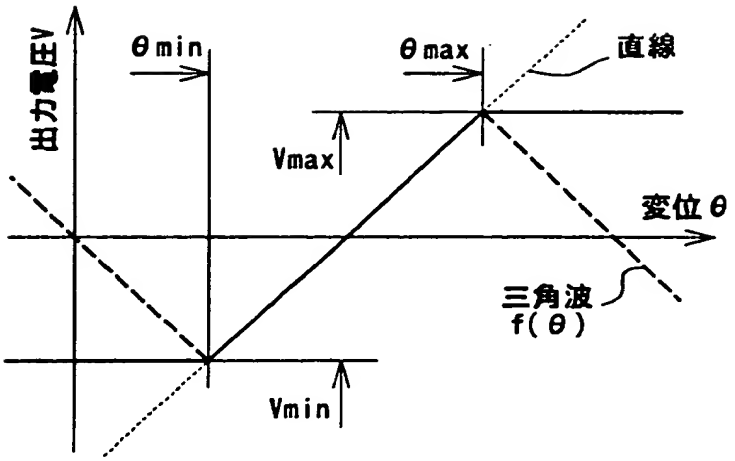
【図５５】


g(φ)出力処理開始
g(φo)出力
S1801
S1802
φ-φo＞π/2
YES
S1803
φo+=π/2
NO
S1804
φo-φ ＞π/2
YES
S1805
φo-=π/2
NO
S1806
50μsec Wait
g(φ)出力
S1807
φo←φ
S1808
g(φ)出力処理終了

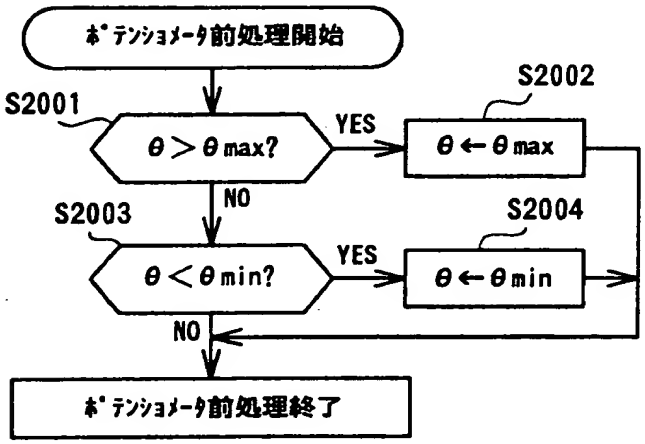
【図５６】


エンコーダ処理開始
S1901
φ←θ変換
S1902
故障フラグ==1
YES
NO
S1903
故障フラグ==2
YES
NO
S1904
g(φ)出力処理
S1905
g(π/4)出力
S1906
g(3π/4)出力
S1907
2相出力?
YES
NO
S1908
U/D信号出力?
YES
NO
S1909
故障フラグ==3
YES
NO
S1910
故障フラグ==4
YES
NO
S1911
g(φ+π/2)出力処理
S1912
g(π/4)出力
S1913
g(3π/4)出力
S1914
故障フラグ==5
YES
NO
S1915
故障フラグ==6
YES
NO
S1916
移動方向?
D
U
S1917
High出力
S1918
Low出力
エンコーダ処理終了

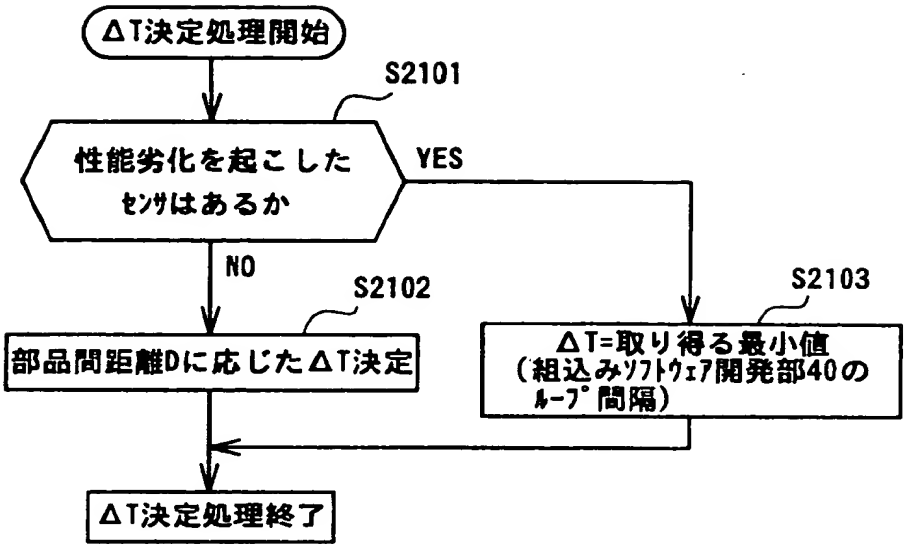
【図５７】


出力電圧V
θmin
θmax
直線
Vmax
変位θ
三角波
f(θ)
Vmin

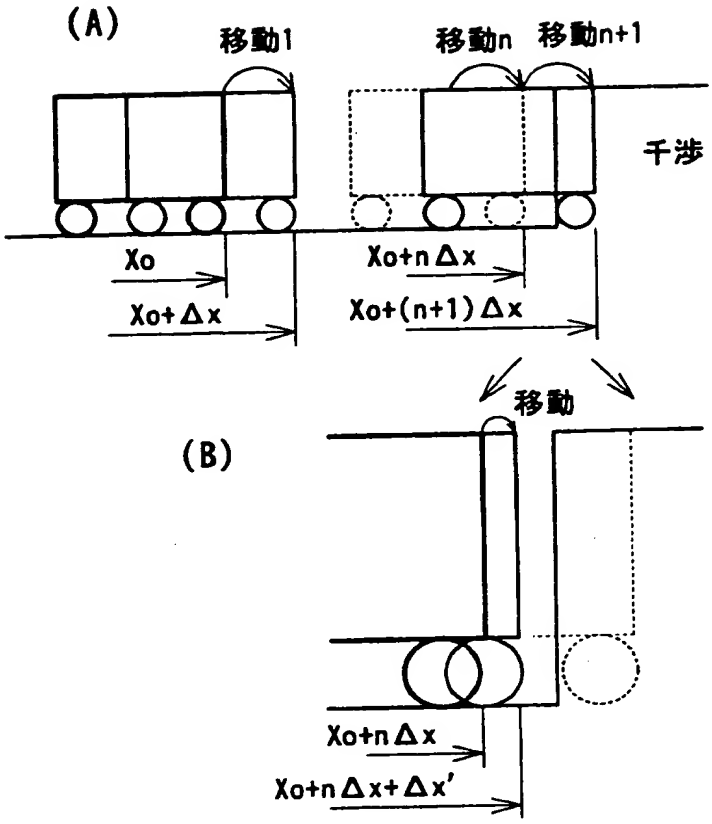
【図５８】


ポテンショメータ前処理開始
S2001
θ＞θmax?
YES
S2002
θ←θmax
NO
S2003
θ＜θmin?
YES
S2004
θ←θmin
NO
ポテンショメータ前処理終了

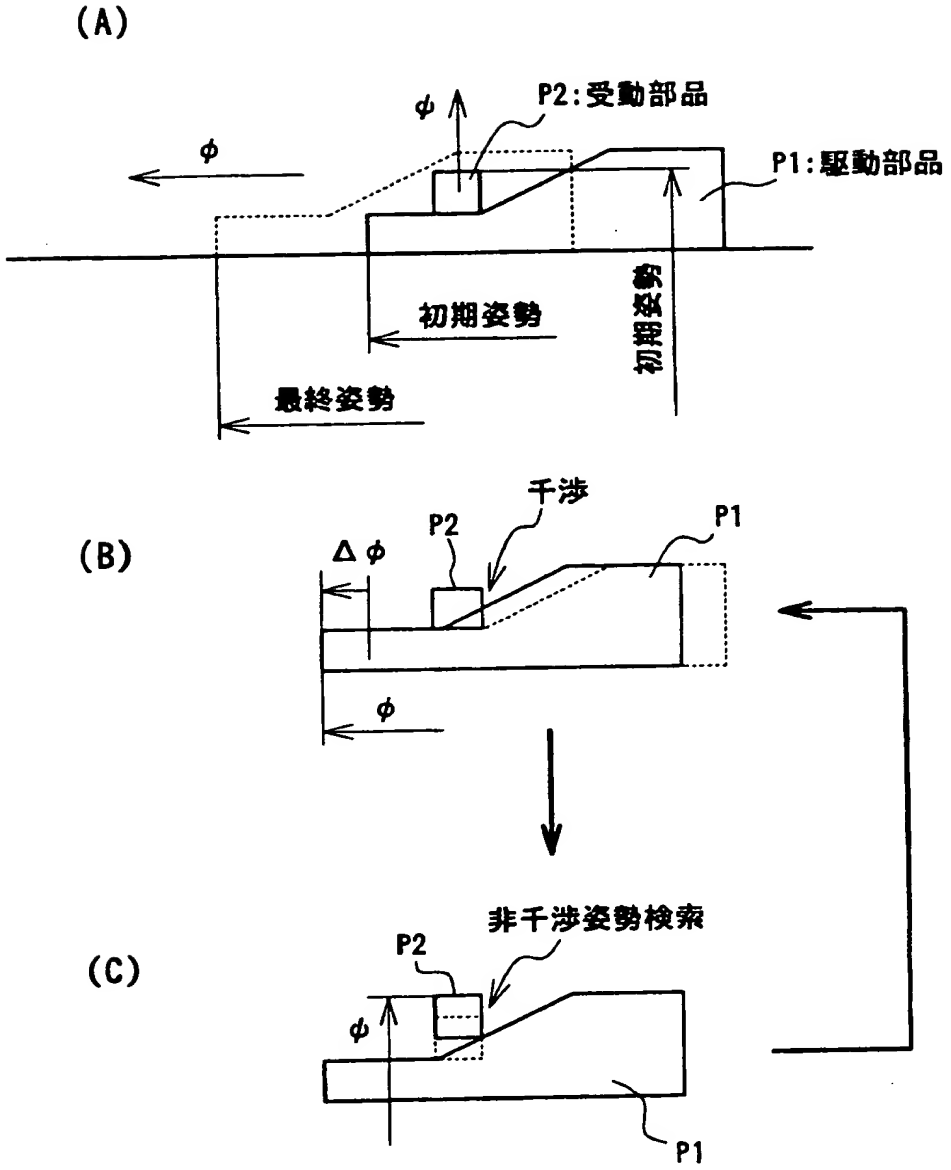
【図５９】


ΔT決定処理開始
S2101
性能劣化を起こした
ｾﾝｻはあるか
YES
NO
S2102
S2103
部品間距離Dに応じたΔT決定
ΔT=取り得る最小値
(組込みｿﾌﾄｳｪｱ開発部40の
ﾙｰﾌﾟ間隔)
ΔT決定処理終了

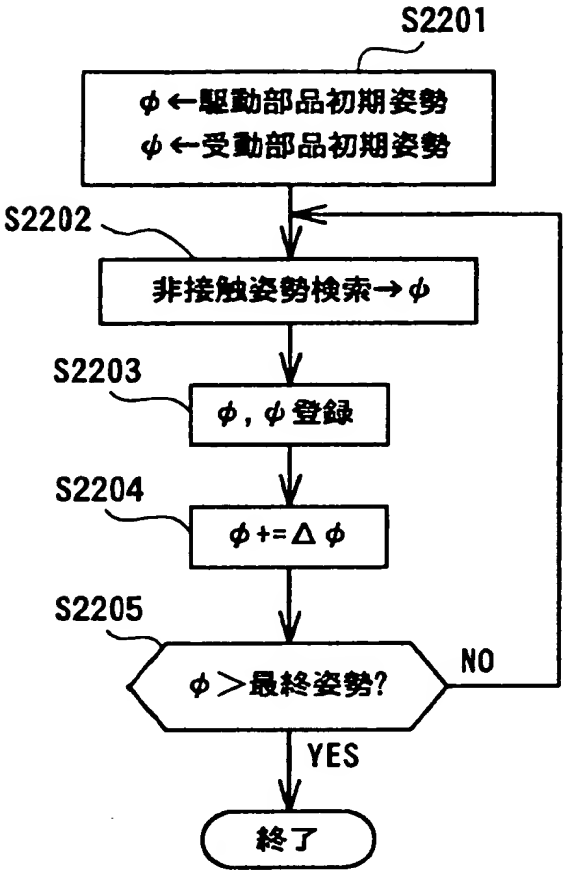
【図６０】



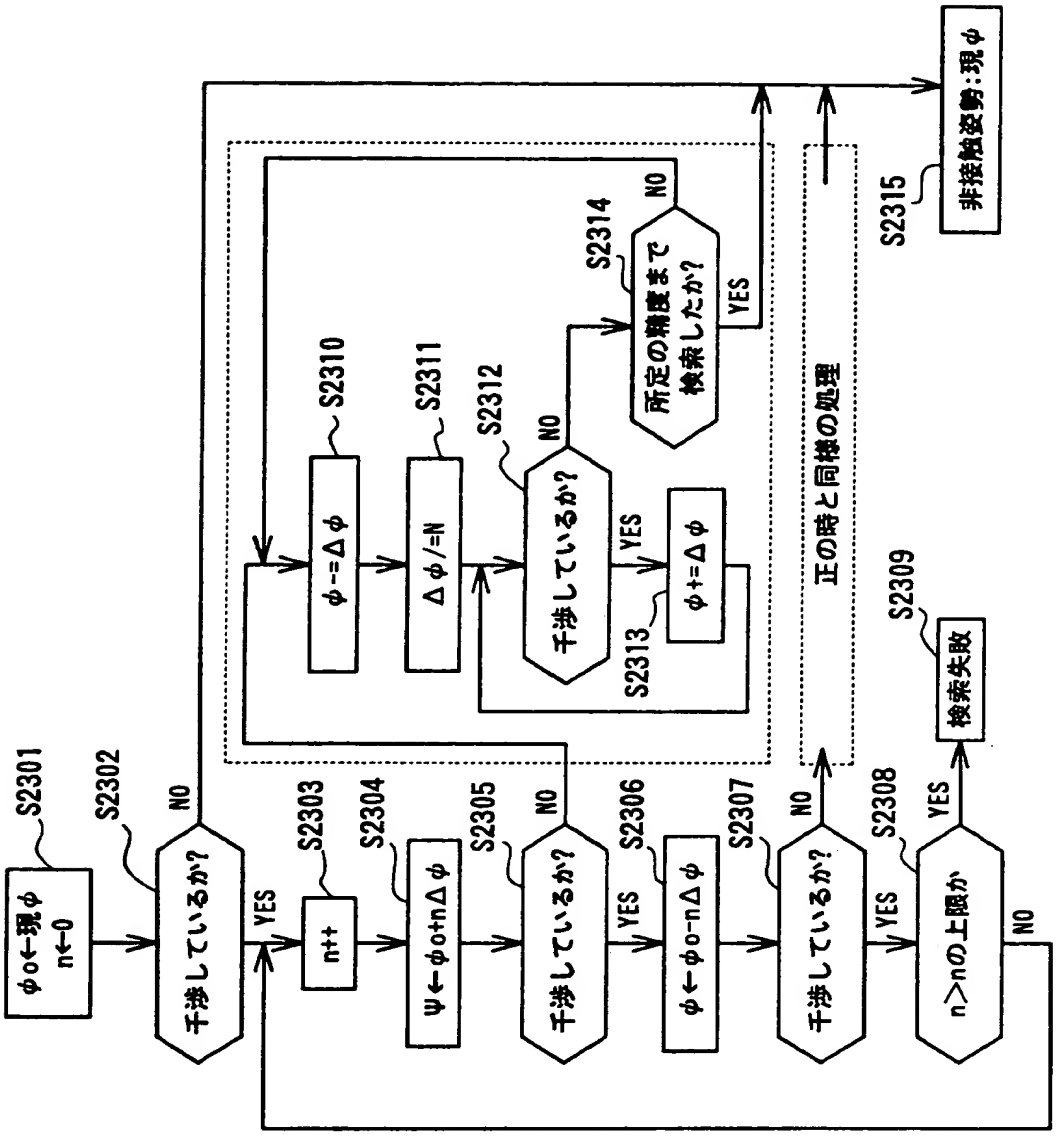
【図６１】


(A)
φ
P2:受動部品
φ
P1:駆動部品
初期姿勢
初期姿勢
最終姿勢
(B)
千渉
P2
P1
Δφ
φ
非千渉姿勢検索
(C)
P2
φ
P1

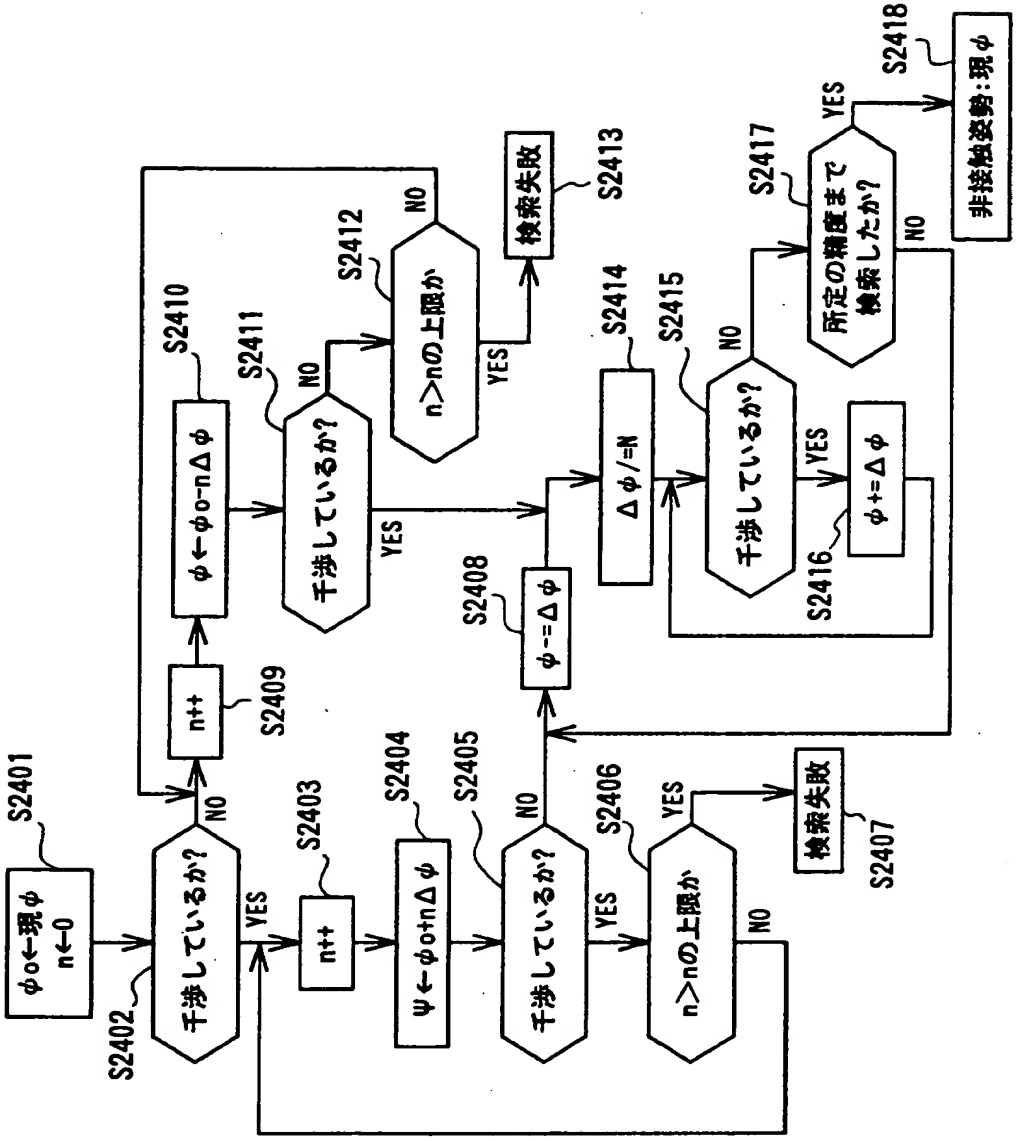
【図６２】


S2201
φ←駆動部品初期姿勢
φ←受動部品初期姿勢
S2202
非接触姿勢検索→φ
S2203
φ，φ登録
S2204
φ+=Δφ
S2205
φ＞最終姿勢?
NO
YES
終了

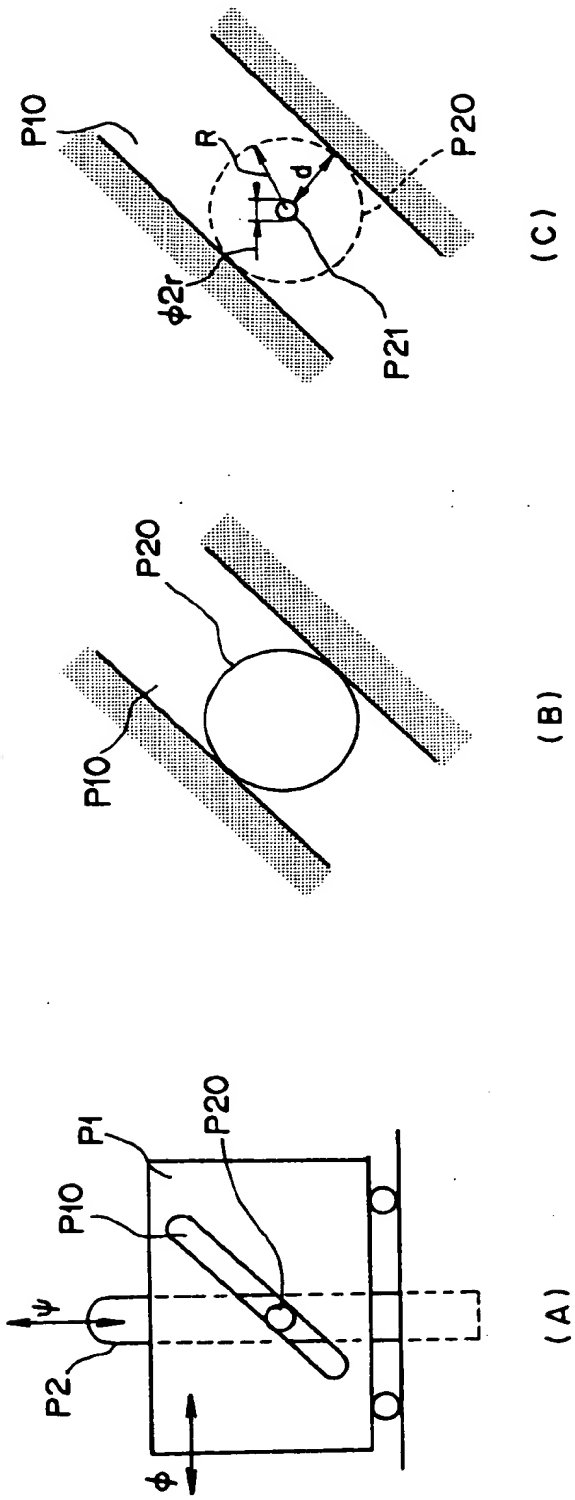
【図６３】


S2301
φo←現φ
n←0
S2302
干渉しているか?
NO
YES
S2303
n++
S2304
Ψ←φo+nΔφ
S2305
干渉しているか?
NO
YES
S2306
φ←φo-nΔφ
S2307
干渉しているか?
NO
YES
S2308
n>nの上限か
YES
NO
S2309
検索失敗
S2310
φ-=Δφ
S2311
Δφ/=N
S2312
干渉しているか?
NO
YES
S2313
φ+=Δφ
S2314
所定の精度まで検索したか?
NO
YES
S2315
非接触姿勢:現φ
正の時と同様の処理

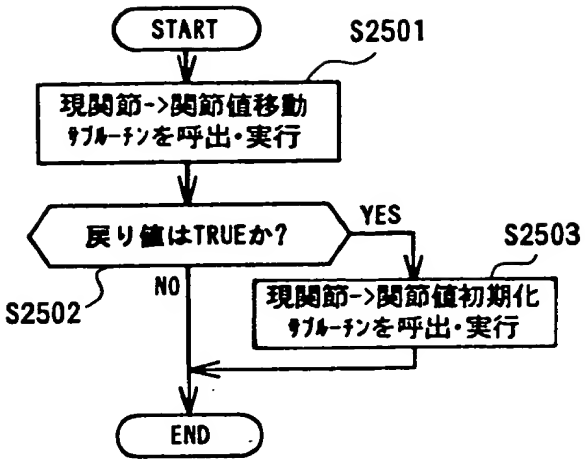
【図６４】


S2401
φo←現φ
n←0
S2402
干渉しているか?
NO
YES
S2403
n++
S2404
Ψ←φo+nΔφ
S2405
干渉しているか?
NO
YES
S2406
n>nの上限か
YES
NO
検索失敗
S2407
S2408
φ-=Δφ
S2409
n++
S2410
φ←φo-nΔφ
S2411
干渉しているか?
NO
YES
S2412
n>nの上限か
NO
YES
検索失敗
S2413
S2414
Δφ/=N
S2415
干渉しているか?
NO
YES
S2416
φ+=Δφ
S2417
所定の精度まで検索したか?
YES
NO
S2418
非接触姿勢:現φ

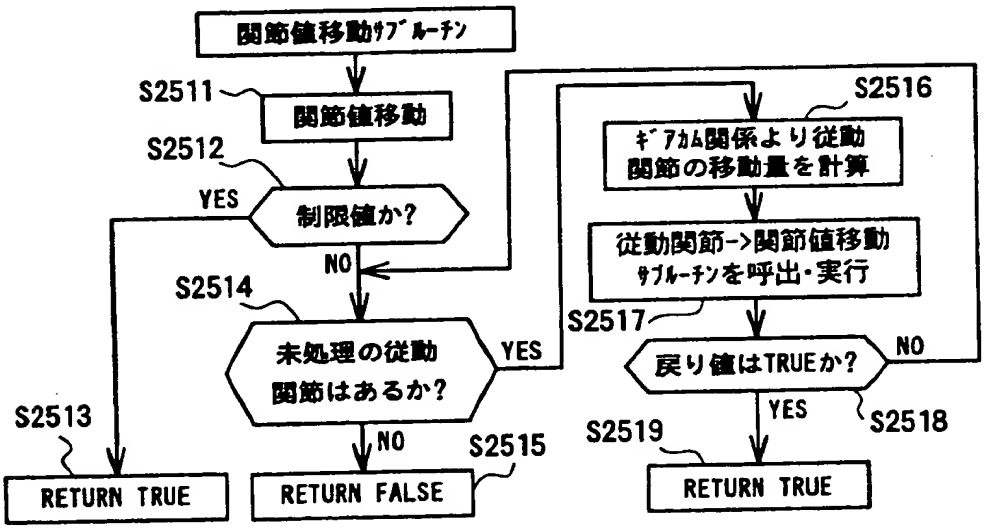
【図６５】



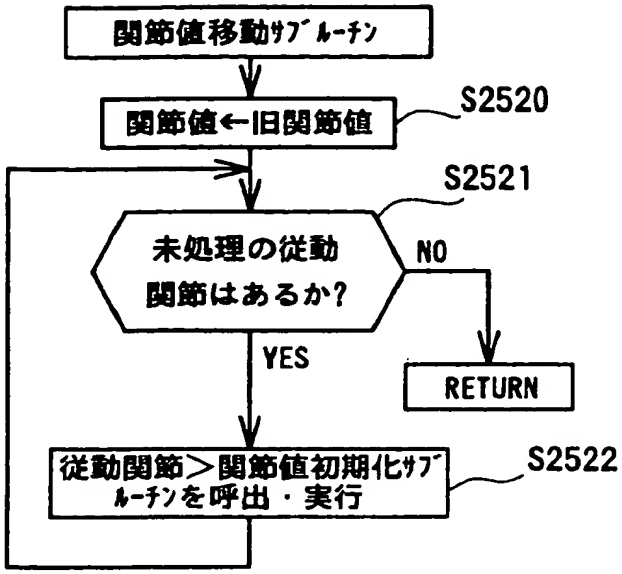
【図６６】


START
S2501
現関節->関節値移動
ｻﾌﾞﾙｰﾁﾝを呼出・実行
戻り値はTRUEか?
YES
S2503
NO
S2502
現関節->関節値初期化
ｻﾌﾞﾙｰﾁﾝを呼出・実行
END

【図６７】

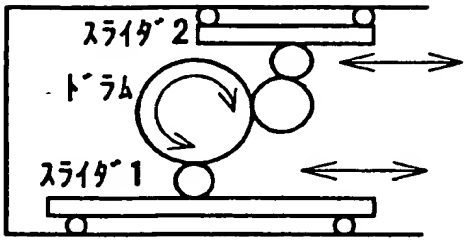

関節値移動ｻﾌﾞﾙｰﾁﾝ
S2511
関節値移動
S2512
YES
制限値か?
NO
S2514
未処理の従動
関節はあるか?
YES
NO
S2513
RETURN TRUE
S2515
RETURN FALSE
S2516
ｷﾞｱｶﾑ関係より従動
関節の移動量を計算
従動関節->関節値移動
ｻﾌﾞﾙｰﾁﾝを呼出・実行
S2517
戻り値はTRUEか?
NO
YES
S2518
S2519
RETURN TRUE

【図６８】

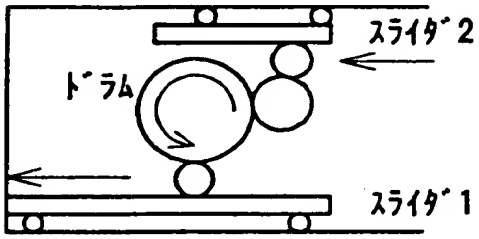

関節値移動ｻﾌﾞﾙｰﾁﾝ
関節値←旧関節値
S2520
S2521
未処理の従動
関節はあるか?
NO
YES
RETURN
従動関節＞関節値初期化ｻﾌﾞ
ﾙｰﾁﾝを呼出・実行
S2522

【図６９】

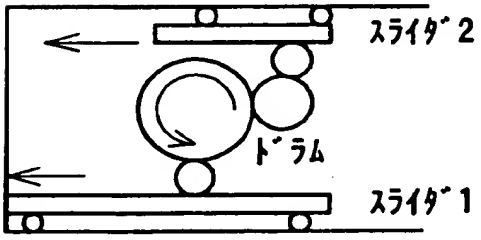
(A)



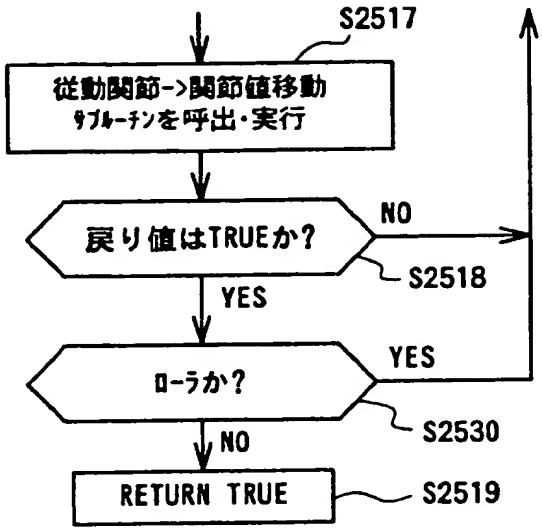
(B)



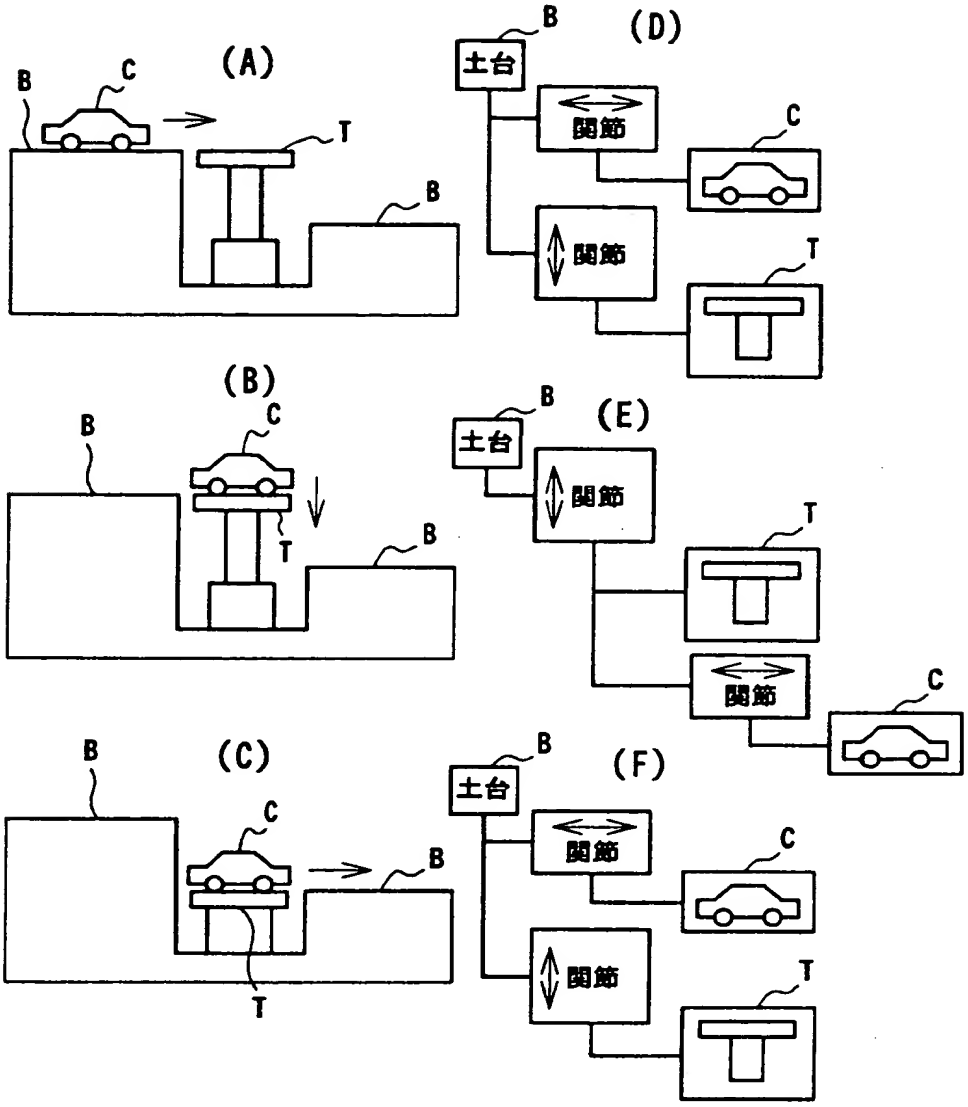
(C)



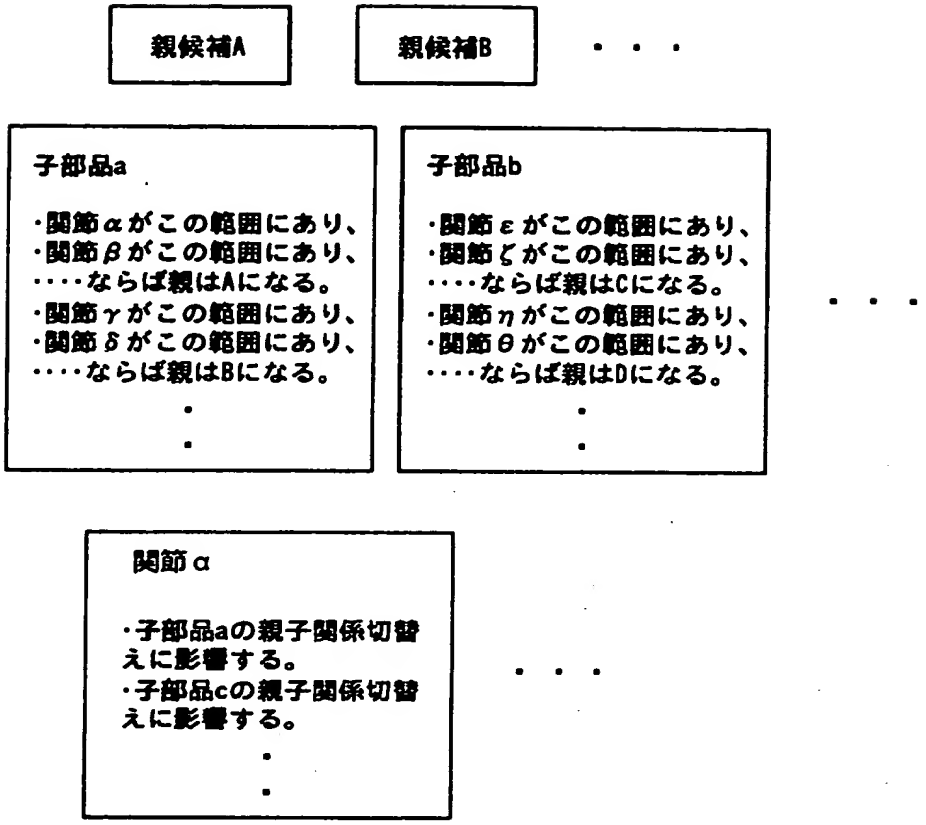
【図７０】


S2517
従動関節->関節値移動
サブルーチンを呼出・実行
戻り値はTRUEか?
NO
S2518
YES
ローラか?
YES
S2530
NO
RETURN TRUE
S2519

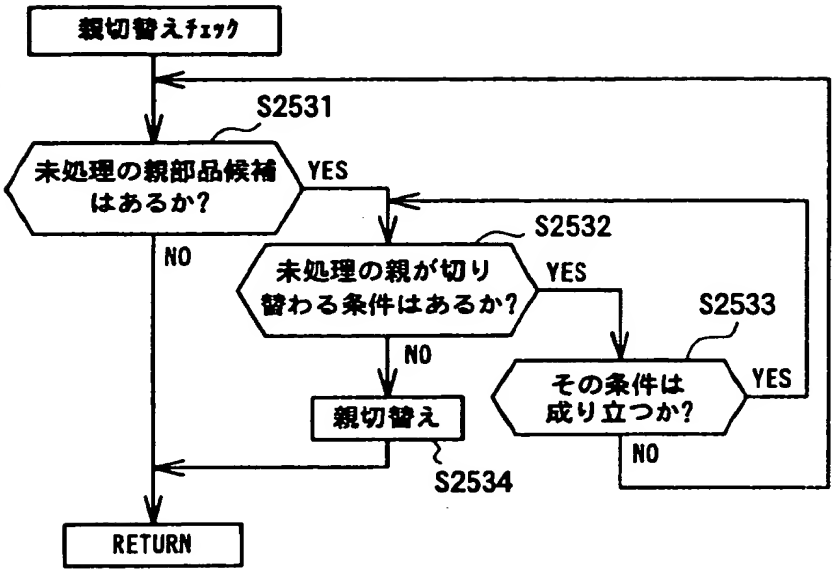
【図７１】



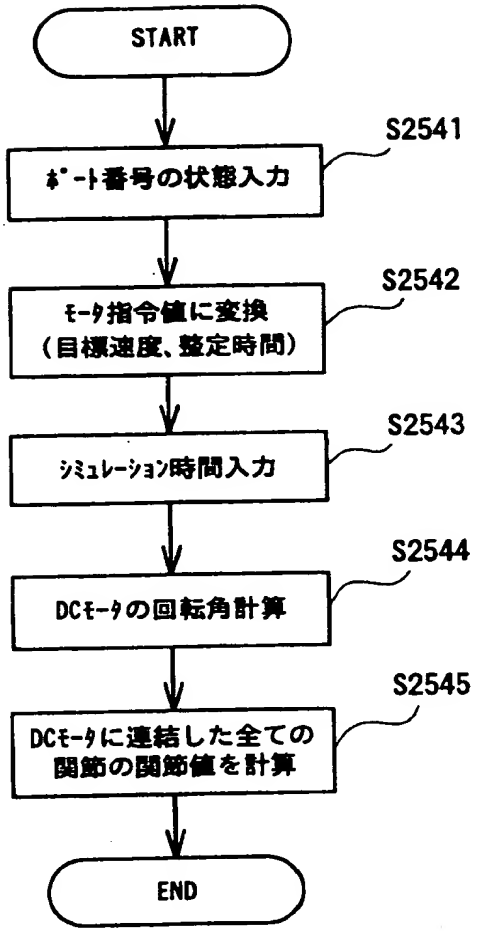
【図７２】


親候補A
親候補B
・・・
子部品a
・関節αがこの範囲にあり、
・関節βがこの範囲にあり、
・・・・ならば親はAになる。
・関節γがこの範囲にあり、
・関節δがこの範囲にあり、
・・・・ならば親はBになる。
子部品b
・関節εがこの範囲にあり、
・関節ζがこの範囲にあり、
・・・・ならば親はCになる。
・関節ηがこの範囲にあり、
・関節θがこの範囲にあり、
・・・・ならば親はDになる。
関節α
・子部品aの親子関係切替えに影響する。
・子部品cの親子関係切替えに影響する。

【図７３】


親切替えチェック
S2531
未処理の親部品候補はあるか?
YES
NO
S2532
未処理の親が切り替わる条件はあるか?
YES
NO
S2533
その条件は成り立つか?
YES
NO
親切替え
S2534
RETURN

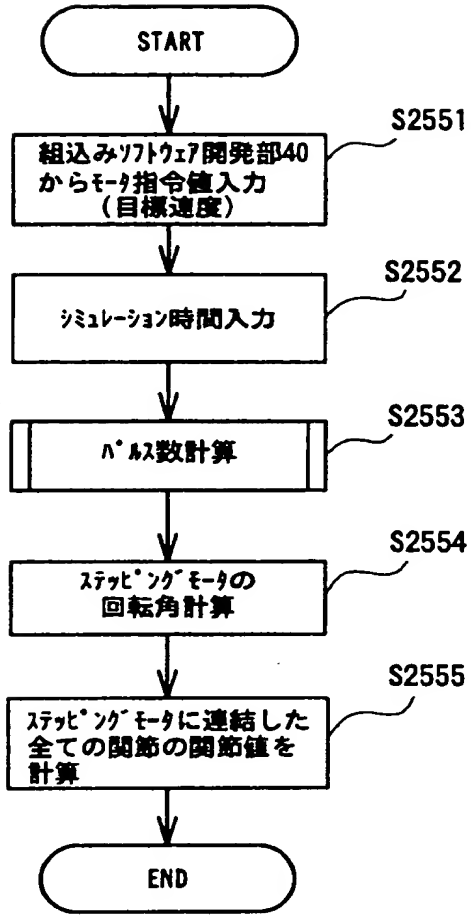
【図７４】



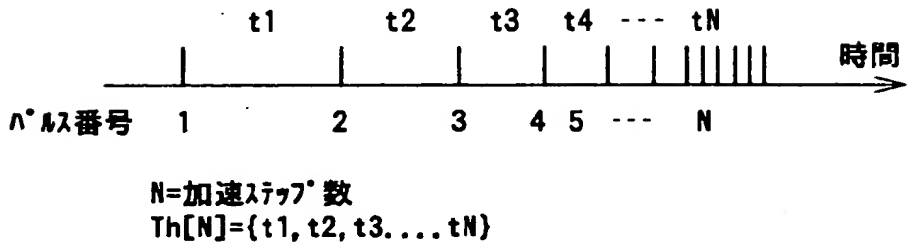
【図７５】

	ﾎﾟｰﾄ番号 11	ﾎﾟｰﾄ番号 12	目標速度 [deg / s]	整定時間 [ms]
ﾋﾞｯﾄ列に対応するﾓｰﾀ指令	0	0	0	0
	0	1	+100	500
	1	0	-100	500
	1	1	0	0

【図７６】

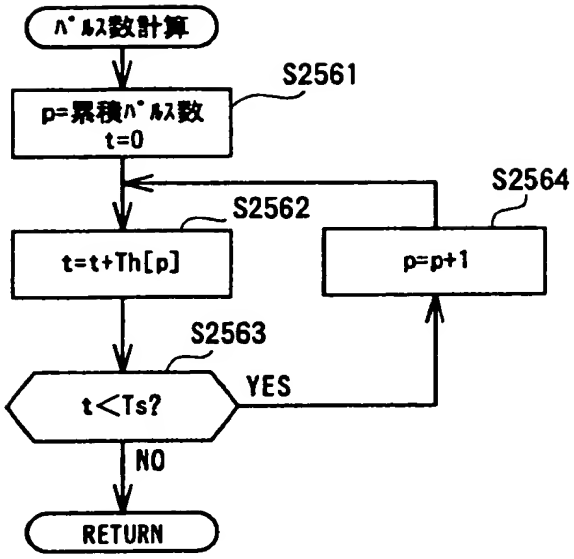


【図７７】

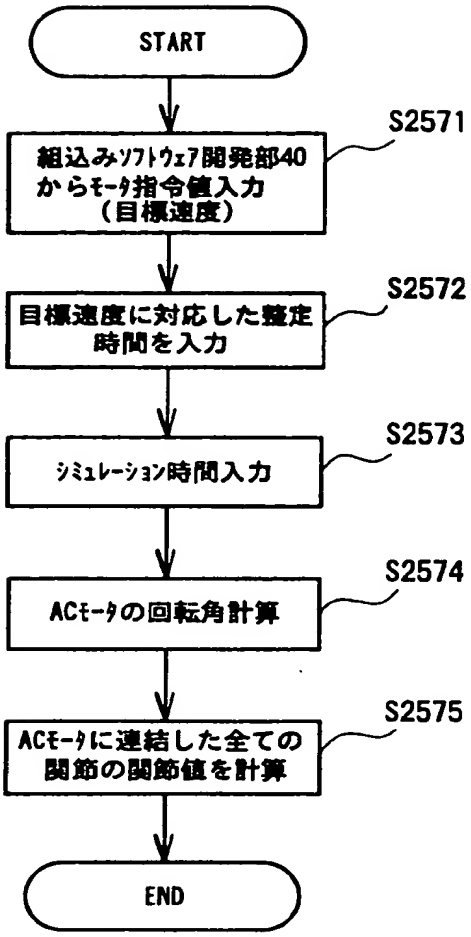


N=加速ｽﾃｯﾌﾟ数
Th[N]={t1, t2, t3. . . . tN}

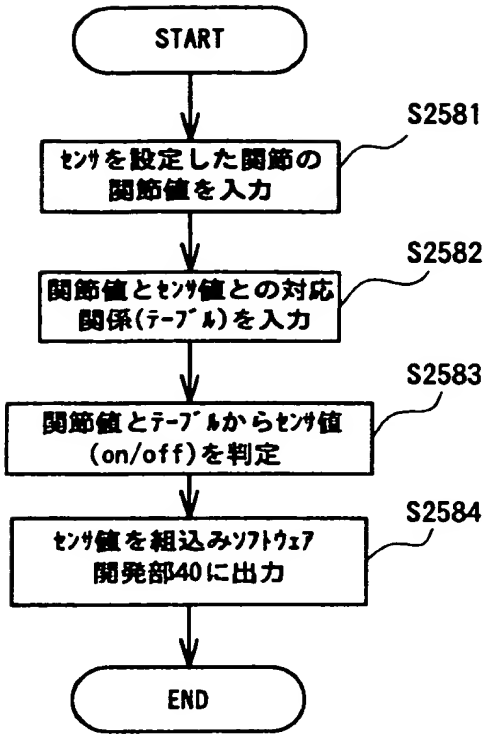
【図７８】


ﾊﾟﾙｽ数計算
S2561
p=累積ﾊﾟﾙｽ数
t=0
S2564
S2562
t=t+Th[p]
p=p+1
S2563
t<Ts?
YES
NO
RETURN

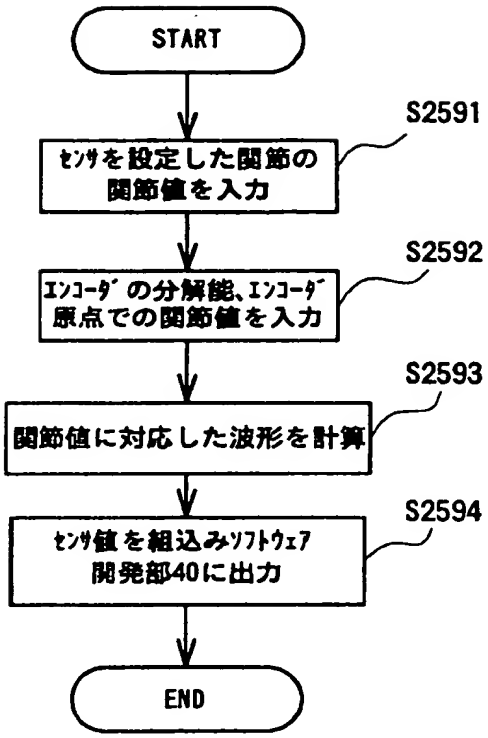
【図７９】


START
組込みソフトウェア開発部40からモータ指令値入力（目標速度）
S2571
目標速度に対応した整定時間を入力
S2572
シミュレーション時間入力
S2573
ACモータの回転角計算
S2574
ACモータに連結した全ての関節の関節値を計算
S2575
END

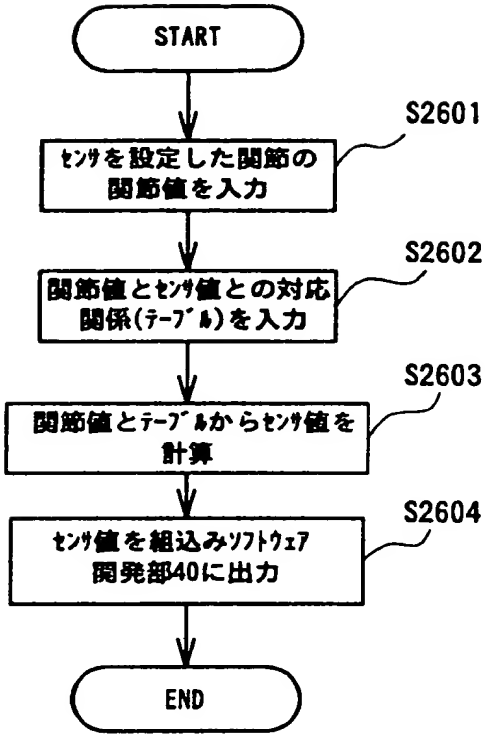
【図８０】


START
S2581
ｾﾝｻを設定した関節の
関節値を入力
S2582
関節値とｾﾝｻ値との対応
関係(ﾃｰﾌﾞﾙ)を入力
S2583
関節値とﾃｰﾌﾞﾙからｾﾝｻ値
(on/off)を判定
S2584
ｾﾝｻ値を組込みｿﾌﾄｳｪｱ
開発部40に出力
END

【図８１】


START
S2591
ｾﾝｻを設定した関節の
関節値を入力
S2592
ｴﾝｺｰﾀﾞの分解能、ｴﾝｺｰﾀﾞ
原点での関節値を入力
S2593
関節値に対応した波形を計算
S2594
ｾﾝｻ値を組込みｿﾌﾄｳｪｱ
開発部40に出力
END

【図８２】


START
S2601
ｾﾝｻを設定した関節の
関節値を入力
S2602
関節値とｾﾝｻ値との対応
関係(ﾃｰﾌﾞﾙ)を入力
S2603
関節値とﾃｰﾌﾞﾙからｾﾝｻ値を
計算
S2604
ｾﾝｻ値を組込みｿﾌﾄｳｪｱ
開発部40に出力
END

【書類名】　　要約書

【要約】

【課題】機構の設計とその機構を制御するための組込みソフトウェアの開発とをコンカレントに遂行できるようにして、その組込みソフトウェアの開発の効率化をはかり、開発期間の短縮，工数の削減を実現する。

【解決手段】機構を３次元的に設計する機構設計部２０と、その機構を３次元機構モデルとして内部に構築され機構の動作をシミュレートする３次元機構モデルシミュレーション部３０と、機構の設計と並行して機構の動作を制御するための制御プログラムを開発する組込みソフトウェア開発部４０と、設計データを３次元機構モデルに動的に反映させるべく機構設計部２０から３次元機構モデルシミュレーション部３０へ入力する第１インタフェース部５０と、３次元機構モデルシミュレーション部３０と組込みソフトウェア開発部４０との同期をとりながらデータ送受を行なう第２インタフェース部５１，５２とがそなられている。

【選択図】　　図１

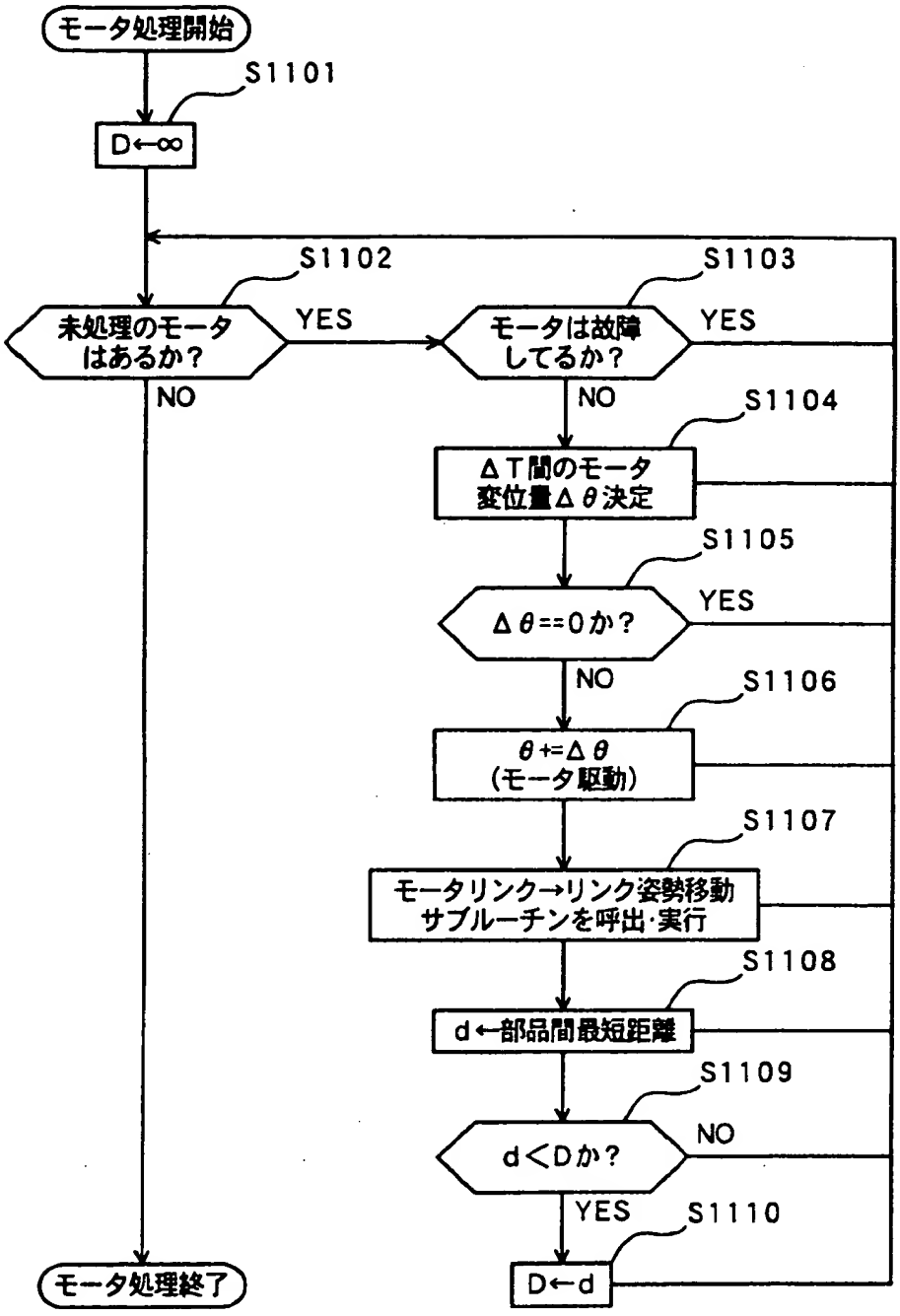
出証特２０００－３０８０２４８

出　願　人　履　歴　情　報

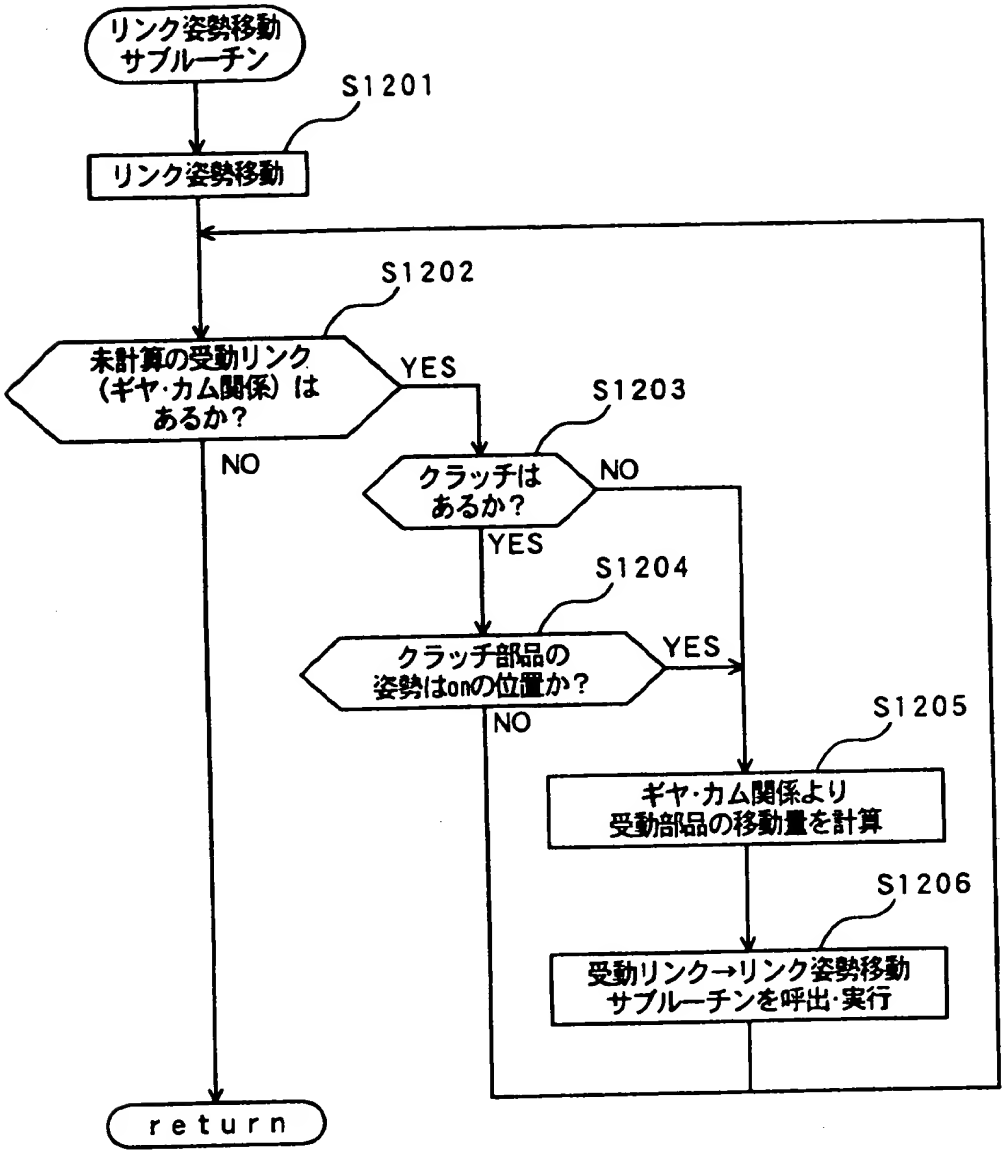
識別番号　　〔０００００５２２３〕

１．変更年月日　　１９９６年　３月２６日
［変更理由］　住所変更
住　所　　神奈川県川崎市中原区上小田中４丁目１番１号
氏　名　　富士通株式会社

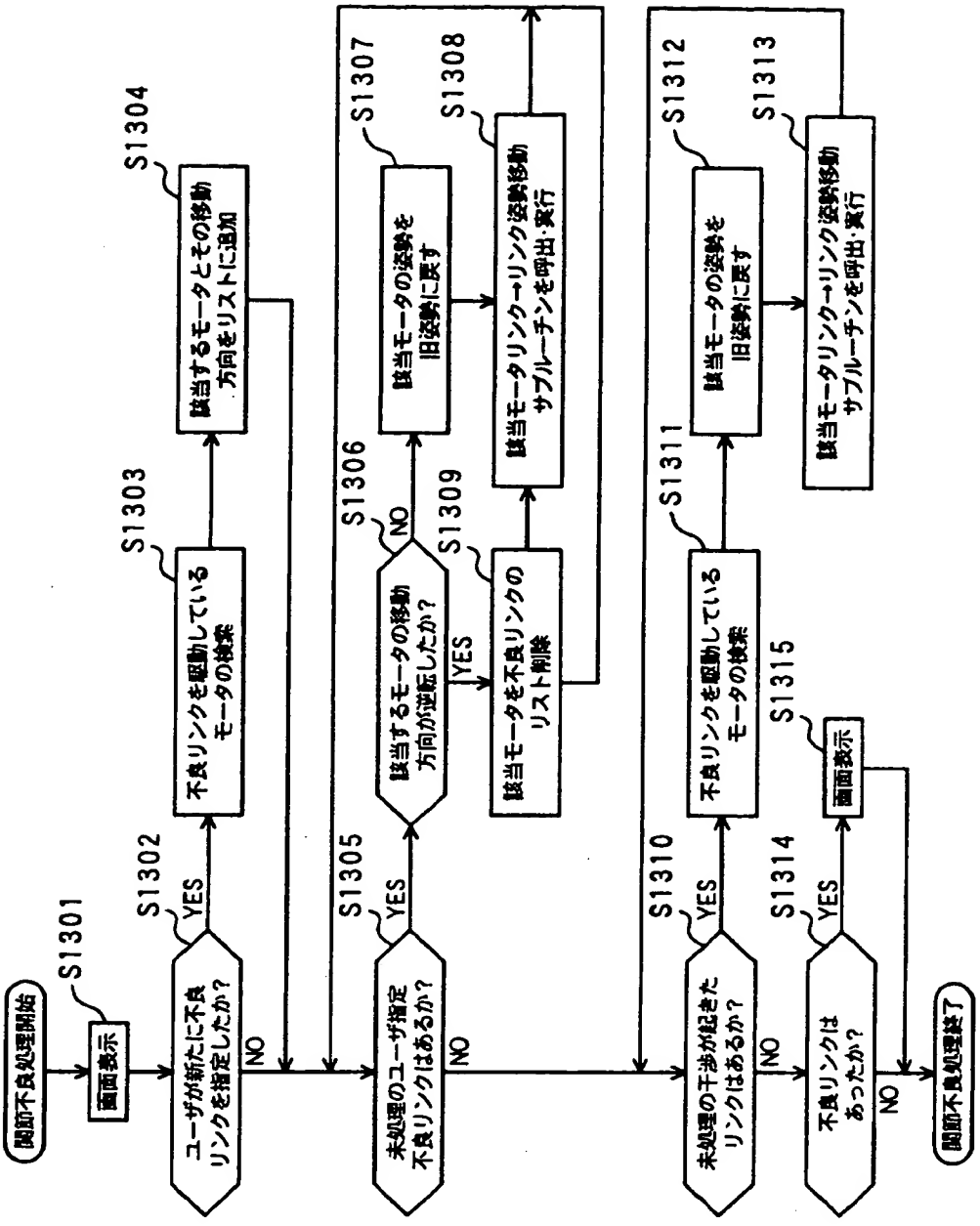
【図４６】


モータ処理開始
S1101
D←∞
S1102
未処理のモータはあるか？
YES
NO
S1103
モータは故障してるか？
YES
NO
S1104
ΔT間のモータ変位量Δθ決定
S1105
Δθ==0か？
YES
NO
S1106
θ+=Δθ (モータ駆動)
S1107
モータリンク→リンク姿勢移動サブルーチンを呼出・実行
S1108
d←部品間最短距離
S1109
d＜Dか？
NO
YES
S1110
D←d
モータ処理終了

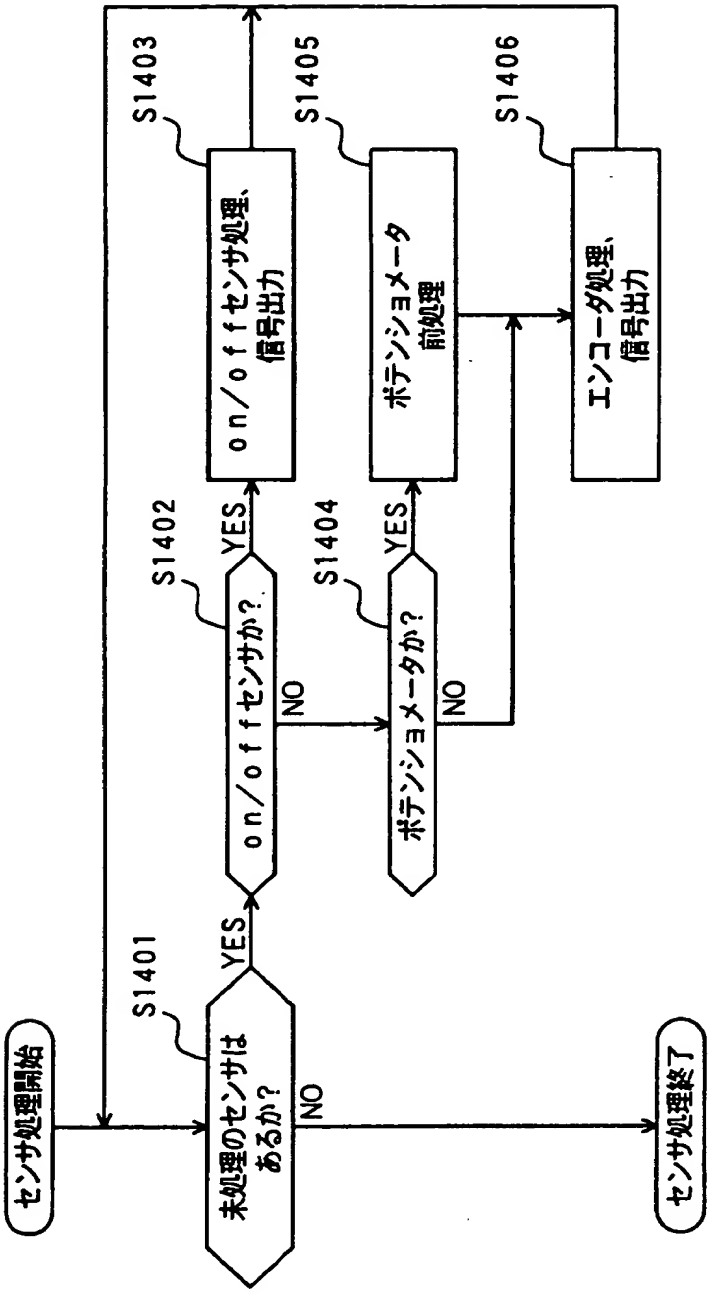
【図４７】


リンク姿勢移動
サブルーチン
S1201
リンク姿勢移動
S1202
未計算の受動リンク
(ギヤ･カム関係) は
あるか?
YES
NO
S1203
クラッチは
あるか?
NO
YES
S1204
クラッチ部品の
姿勢はonの位置か?
YES
NO
S1205
ギヤ･カム関係より
受動部品の移動量を計算
S1206
受動リンク→リンク姿勢移動
サブルーチンを呼出･実行
return

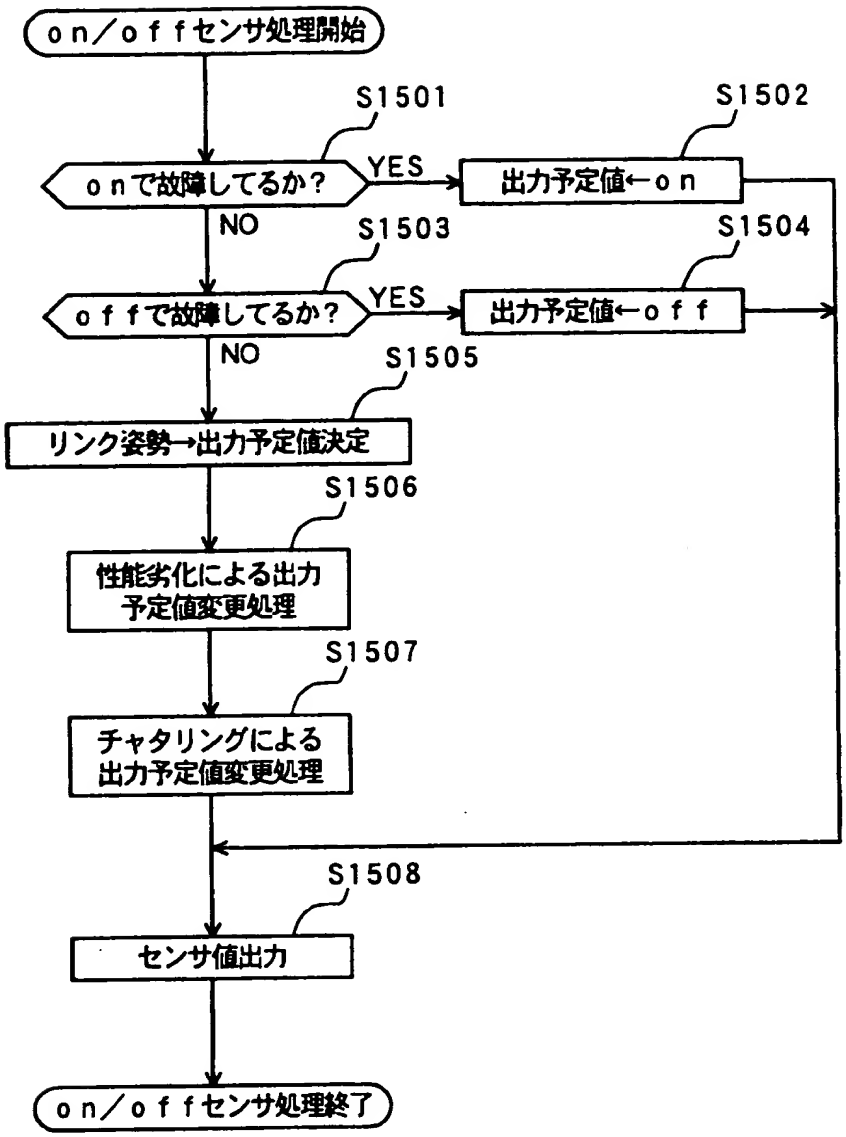
【図４８】


関節不良処理開始
S1301
画面表示
S1302
ユーザが新たに不良リンクを指定したか？
YES
NO
S1303
不良リンクを駆動しているモータの検索
S1304
該当するモータとその移動方向をリストに追加
S1305
未処理のユーザ指定不良リンクはあるか？
YES
NO
S1306
該当するモータの移動方向が逆転したか？
NO
YES
S1307
該当モータの姿勢を旧姿勢に戻す
S1308
該当モータリンク→リンク姿勢移動サブルーチンを呼出・実行
S1309
該当モータを不良リンクのリスト削除
S1310
未処理の干渉が起きたリンクはあるか？
YES
NO
S1311
不良リンクを駆動しているモータの検索
S1312
該当モータの姿勢を旧姿勢に戻す
S1313
該当モータリンク→リンク姿勢移動サブルーチンを呼出・実行
S1314
不良リンクはあったか？
YES
NO
S1315
画面表示
関節不良処理終了

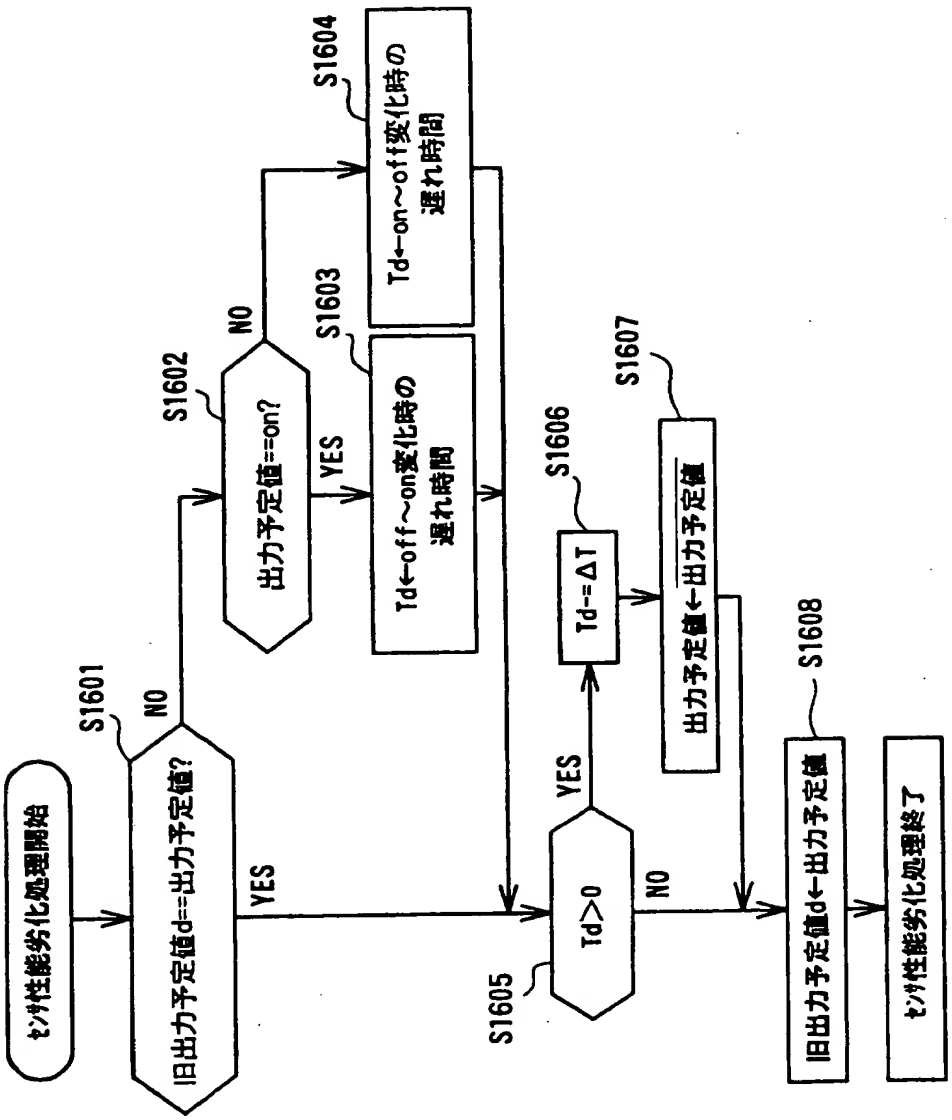
【図４９】


センサ処理開始
S1401
未処理のセンサはあるか？
YES
NO
センサ処理終了
S1402
on／offセンサか？
YES
NO
S1403
on／offセンサ処理、信号出力
S1404
ポテンショメータか？
YES
NO
S1405
ポテンショメータ前処理
S1406
エンコーダ処理、信号出力

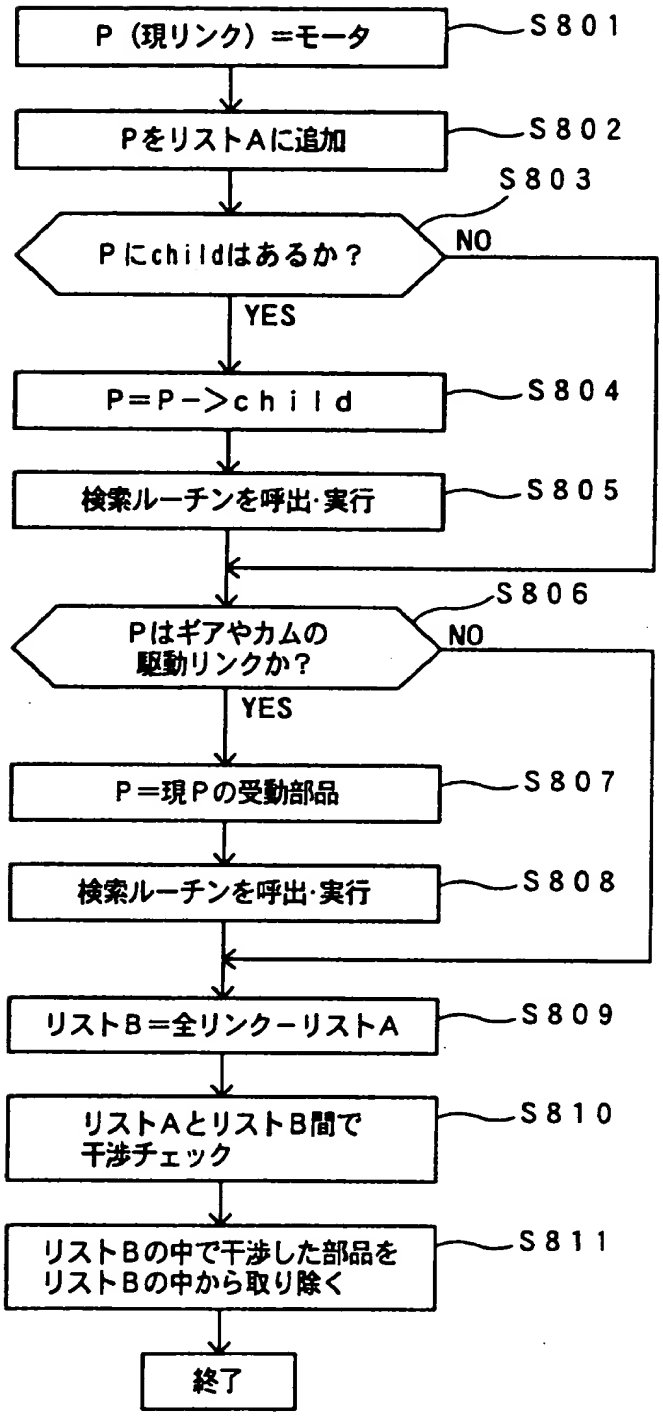
【図５０】


ｏｎ／ｏｆｆセンサ処理開始
S1501
ｏｎで故障してるか？
YES
S1502
出力予定値←ｏｎ
NO
S1503
ｏｆｆで故障してるか？
YES
S1504
出力予定値←ｏｆｆ
NO
S1505
リンク姿勢→出力予定値決定
S1506
性能劣化による出力予定値変更処理
S1507
チャタリングによる出力予定値変更処理
S1508
センサ値出力
ｏｎ／ｏｆｆセンサ処理終了

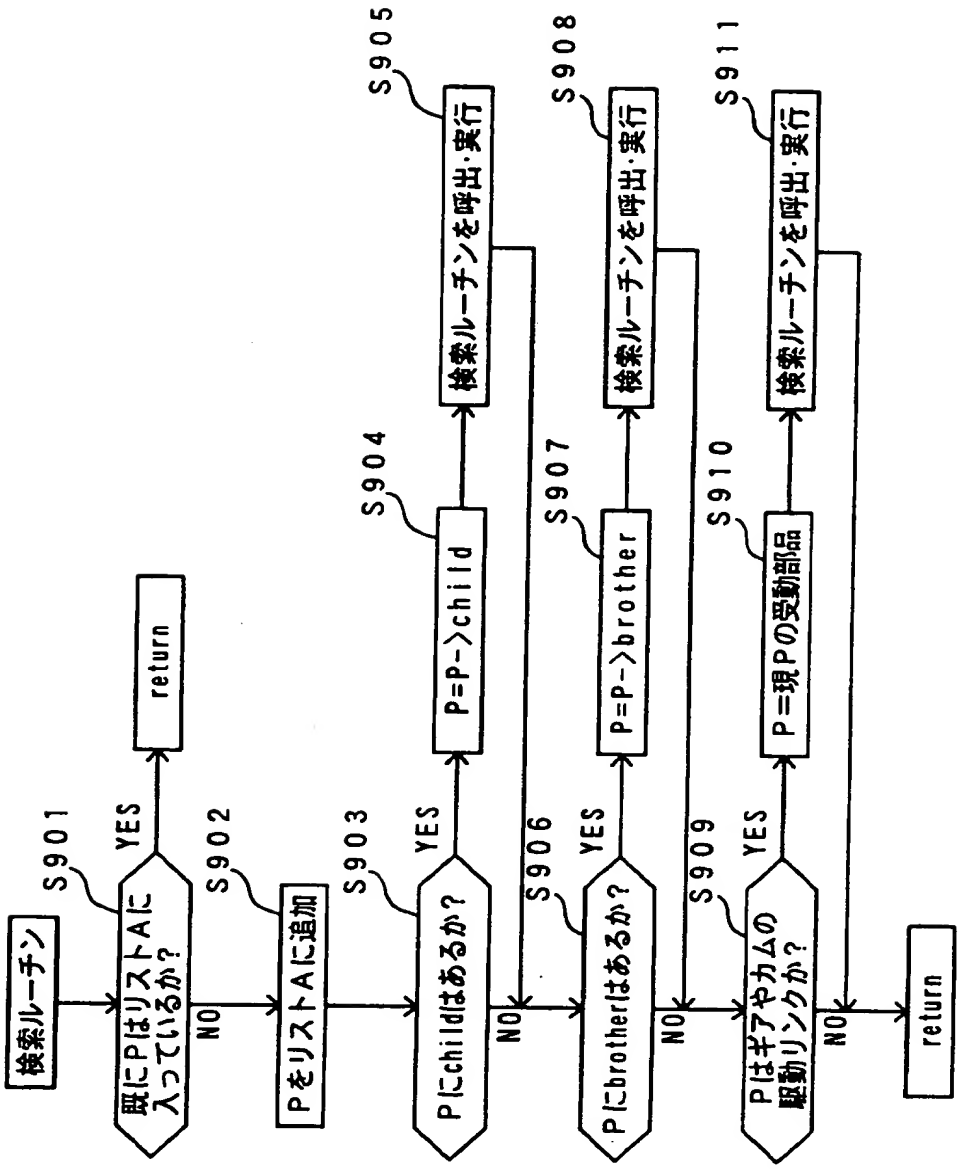
【図５１】


センサ性能劣化処理開始
S1601
旧出力予定値d==出力予定値?
NO
YES
S1602
出力予定値==on?
NO
YES
S1603
Td←off～on変化時の遅れ時間
S1604
Td←on～off変化時の遅れ時間
S1605
Td>0
YES
NO
S1606
Td-=ΔT
S1607
出力予定値←出力予定値
S1608
旧出力予定値d←出力予定値
センサ性能劣化処理終了

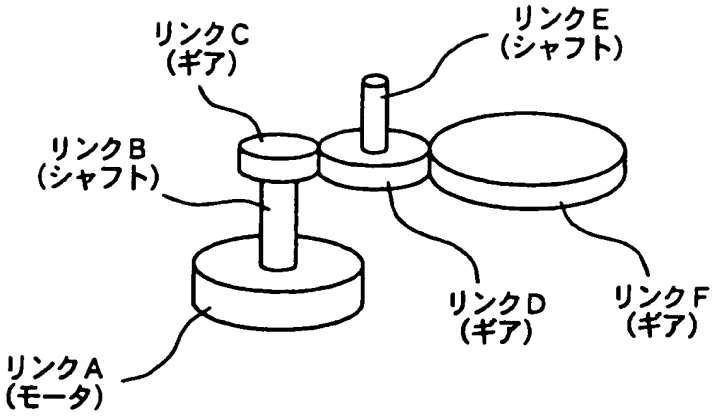
【図４０】


P（現リンク）＝モータ
S801
Pをリストに追加
S802
S803
Pにchildはあるか？
NO
YES
P＝P－＞child
S804
検索ルーチンを呼出・実行
S805
S806
Pはギアやカムの
駆動リンクか？
NO
YES
P＝現Pの受動部品
S807
検索ルーチンを呼出・実行
S808
リストB＝全リンクーリストA
S809
リストAとリストB間で
干渉チェック
S810
リストBの中で干渉した部品を
リストBの中から取り除く
S811
終了

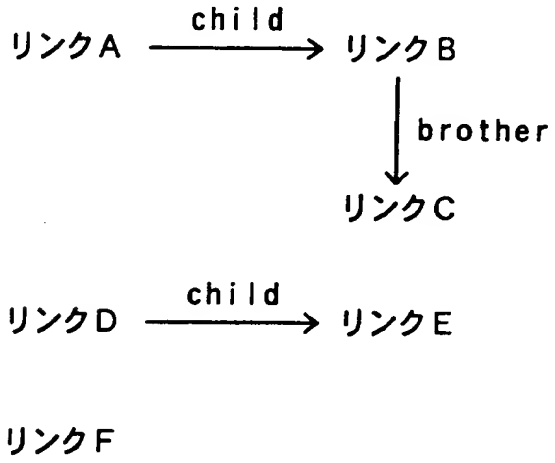
【図４１】


検索ルーチン
S901
既にPはリストAに入っているか？
YES
return
NO
S902
PをリストAに追加
S903
Pにchildはあるか？
YES
S904
P=P->child
S905
検索ルーチンを呼出・実行
NO
S906
Pにbrotherはあるか？
YES
S907
P=P->brother
S908
検索ルーチンを呼出・実行
NO
S909
Pはギアやカムの駆動リンクか？
YES
S910
P=現Pの受動部品
S911
検索ルーチンを呼出・実行
NO
return

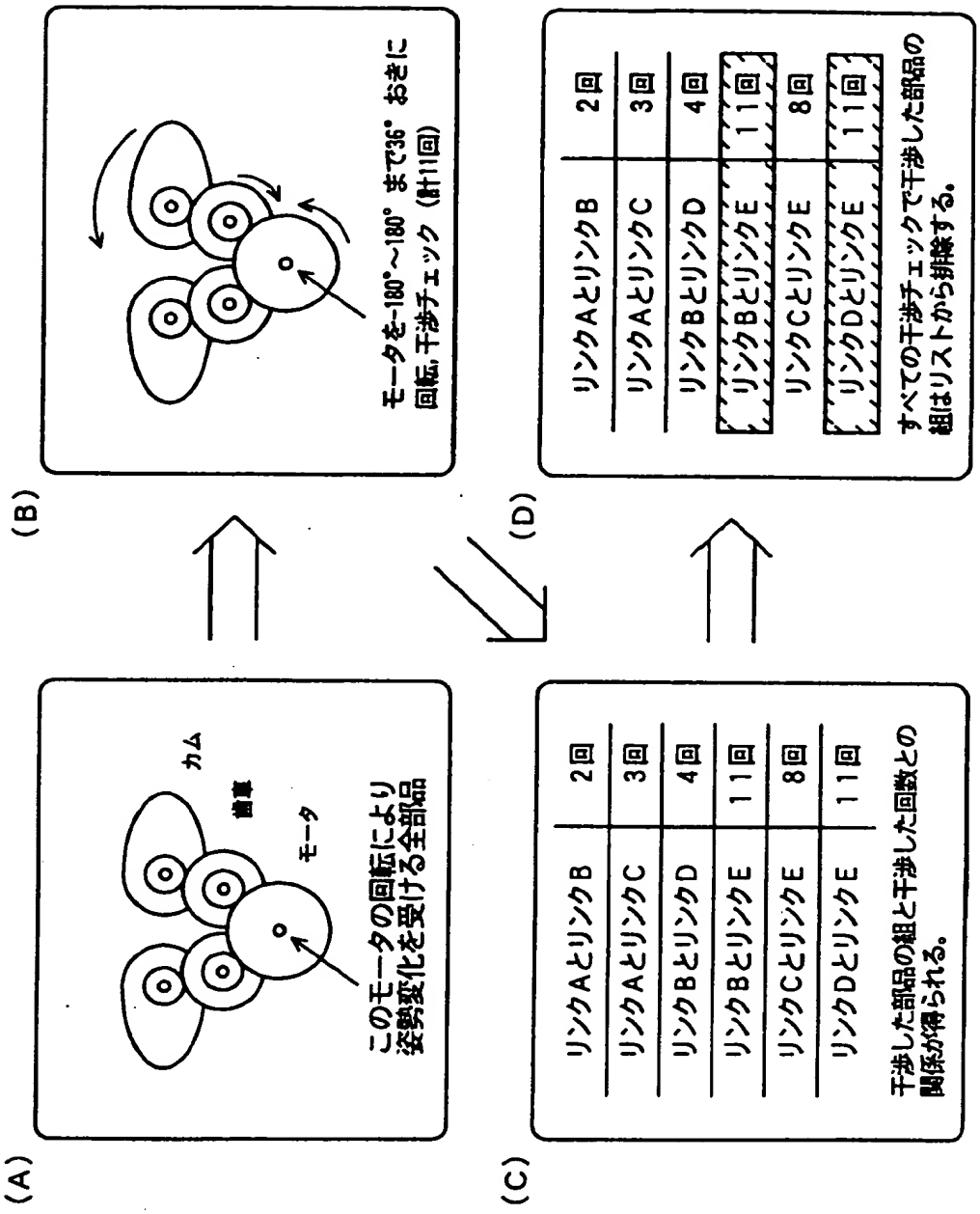
【図４２】


リンクC
(ギア)
リンクE
(シャフト)
リンクB
(シャフト)
リンクD
(ギア)
リンクF
(ギア)
リンクA
(モータ)

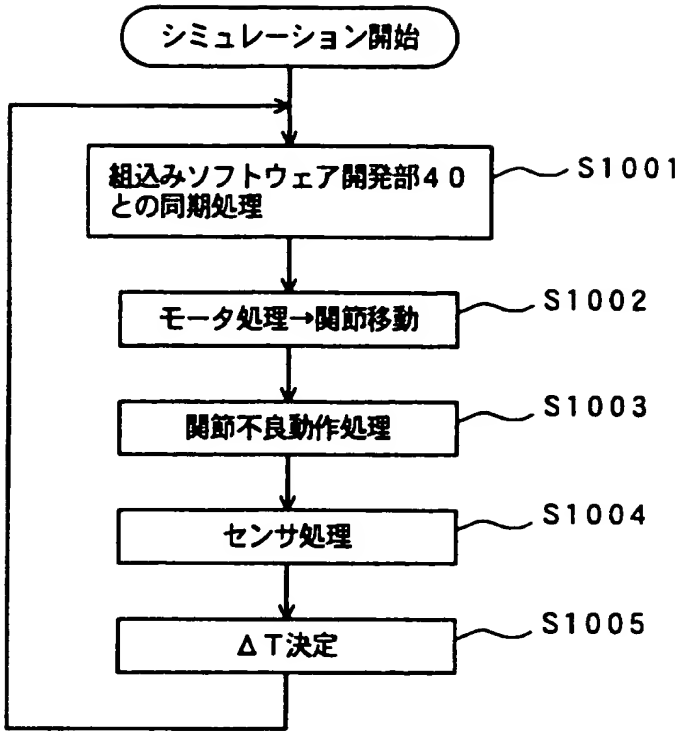
【図４３】


リンクA
child
リンクB
brother
リンクC
リンクD
child
リンクE
リンクF

【図４４】


(A)
このモータの回転により
姿勢変化を受ける全部品
カム
歯車
モータ
(B)
モータを-180°～180°まで36°おきに
回転、干渉チェック（計11回）
(C)
リンクAとリンクB 2回
リンクAとリンクC 3回
リンクBとリンクD 4回
リンクBとリンクE 11回
リンクCとリンクE 8回
リンクDとリンクE 11回
干渉した部品の組と干渉した回数との
関係が得られる。
(D)
リンクAとリンクB 2回
リンクAとリンクC 3回
リンクBとリンクD 4回
リンクBとリンクE 11回
リンクCとリンクE 8回
リンクDとリンクE 11回
すべての干渉チェックで干渉した部品の
組はリストから排除する。

【図４５】


シミュレーション開始
組込みソフトウェア開発部４０との同期処理
S1001
モータ処理→関節移動
S1002
関節不良動作処理
S1003
センサ処理
S1004
ΔT決定
S1005